パソコン教育講座シリーズ

PC-8801mk(マーク)Ⅱの使い方 上巻

PC-8801mkⅡの使い方 上巻

工学博士 松尾三郎 監修

本書はN88-BASICのプログラムの作り方、プリンタやフロッピーディスクの使い方
漢字の表示方法など幅広く、設問形式で解説した独習テキストです。

EDC 電子開発学園

パソコン教育講座シリーズ

PC-8801mkⅡの使い方 上巻

EDC 電子開発学園

PC-8801mkII（マーク）の使い方 上巻

工学博士 松尾三郎 監修

電子開発学園

監修者のことば

ここ数年，パソコン（パーソナル・コンピュータ）の普及には目を見はるものがあります。マイコン（マイクロ・コンピュータ）とかパソコンとか呼び名も不統一なまま世に出たパソコンもあっという間に，小学生はおろか幼稚園の子供からお年寄りまでが，これを操作する時代になりました。

技術革新によって精度の高い超LSIなどの半導体技術が高度に進んだ結果，手のひらに乗るほどのコンピュータが実現し，値段も安くなって一般大衆の商品となりました。

このようにパソコンのハードウェア技術は急速に進歩しましたが，反面，これを使いこなす技術であるソフトウェアの開発はあくまで人の頭脳に頼るところから，需要に十分応えられないのが現状です。コンピュータを生かすも殺すもソフトウェアの成否にかかっています。いくら高性能のパソコンでもソフトウェアがなければ，ただの金属のかたまりに過ぎません。ソフトウェア技術者の能力が高度に発揮されて初めて良質のソフトウェアが作り出されるのです。

私はハードウェア，ソフトウェアという言葉自体が普及していなかった15年前，今日のこのコンピュータ時代の到来を予見し，将来の日本の情報化の一助とするためソフトウェア技術者を養成する電子開発学園を創設しました。全国で８校のコンピュータ専門の専修学校を開き，これまでに２万余人の技術者を世に送り出しました。同時にまた，ソフトウェア技術者を養成するには実務教育が必要であるため，日本電子開発株式会社ならびにソフトウェア・コンサルタント株式会社を設けて教育と実務を直結させる産学協同の実践を図っております。

パソコン全盛の時を迎えソフトウェア技術の啓蒙普及の必要性をますます痛感しています。そこで，これまで蓄積してきた教育現場での経験や技術開発の実績に基づいて，自宅学習でもご理解いただけるような「パソコン教育講座シリーズ」のテキストを刊行しました。「PC-8801mkIIの使い方」はこのシリーズの一環をなすものです。

本書は，発売以来大変ご好評をいただいておりますPC-8800 BASIC入門編，初級編，中級編を全面的に改版の上，より使いやすさをモットーにPC-8801mkII初級用のテキストとして更に内容を充実させたものにしました。必ずや皆様のお役に立つものと信じております。

このテキストをもとに，一日も早くパソコンを自分のものにし，自由自在に使いこなすようになって頂ければ，あなたの人生に新しい世界が開かれることは間違いありません。

1984年12月

松尾三郎

はじめに

最近では，OA化ブームを映してだれでも一度はパソコンに目を向けるようになり，必ず目に（または耳に）するのがBASICです。このBASICは，もともと初心者用の言語として作られたものであり，コンピュータで使われているいろいろな言語の中では最も初心者向きであるといえます。

そのために，BASICは機能性に乏しいと誤解される場合がありますが，実際には命令や機能が多岐にわたり，BASICを十分理解した上で利用すると，使い方によっては非常に高度な処理も可能になります。

しかし，初心者にとっては一つ一つの命令がどんな働きをするのか，または，どんなときにどの命令を使えばよいのか覚えるのはたいへん難しいことです。ましてや，プログラムの作り方や考え方となるとなおさらといえましょう。

そこで，本書は身近なテーマを使って命令の働きやその書き方からプログラムの作り方までを段階的に取り上げた設問を通して，操作手順や実行したときの結果例を掲げて分かりやすく解説しています。

本書は，１～９章と練習問題および付録で構成されていて，その主な内容は次のとおりです。

１章はPC-8801mkIIの基本操作について，２章は初歩的なプログラムの学習，３章は配列の使い方，４章と５章は処理の基本となるテクニックについて，６章はシーケンシャルファイル，７章と８章は関数と漢字について，９章はゲームプログラムと音楽を演奏するプログラムの作り方について説明しています。

なお，各章の執筆は電子開発学園出版局の後藤佐千穂が担当し，全体のとりまとめは同局の山縣恵昭が行いました。

さらに，このテキストは基本的にはPC-8801mkII model30（５インチ・ミニ・フロッピーディスク装置２ドライブを本体に内蔵）のマシンを前提に説明されていますが，プログラムを確認するために使用したパソコンの構成は次のとおりです。

①本体――――――――――――PC-8801mkII model30

②CRTディスプレイ――――――PC-8853（高解像度のカラーディスプレイ）

③プリンタ――――――――――PC-8822（漢字ドットプリンタ）

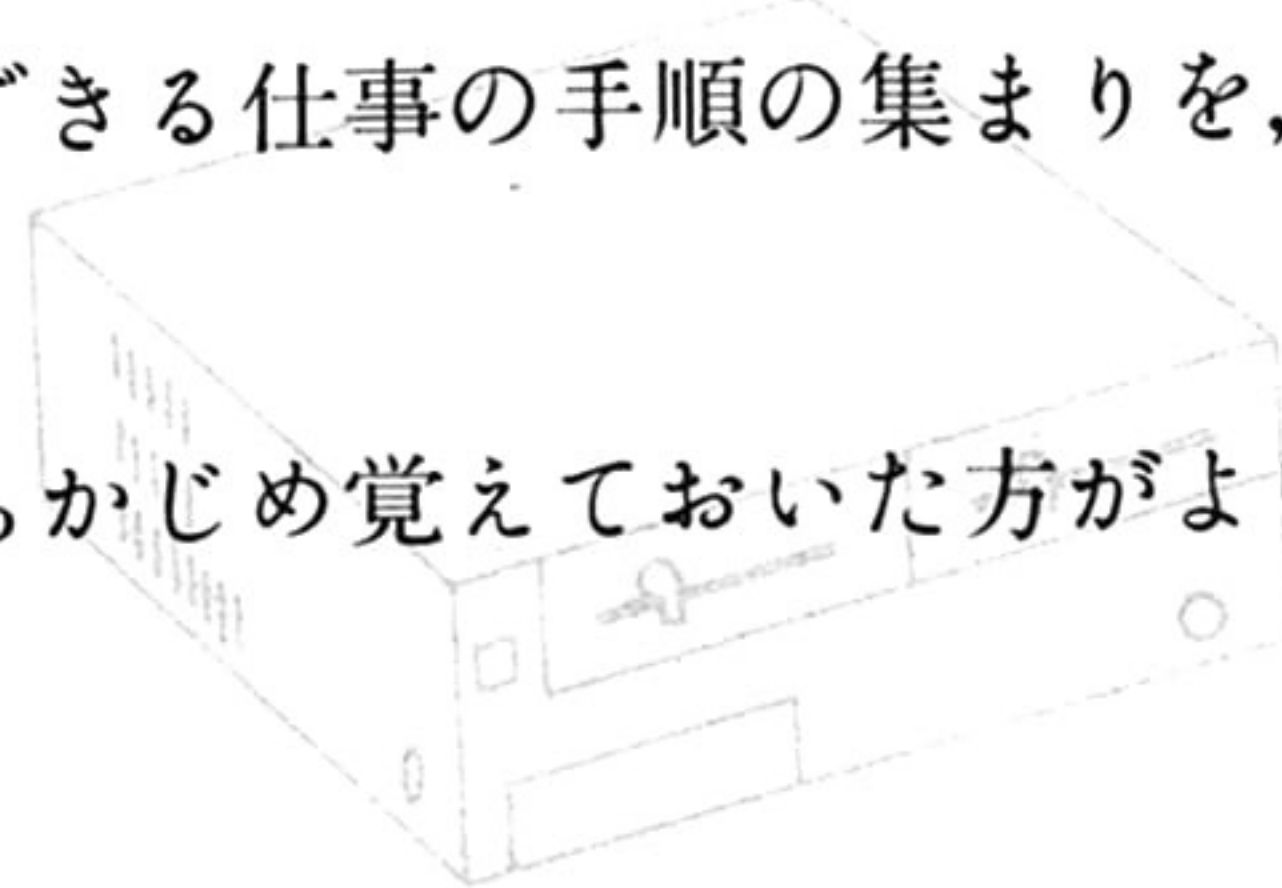
本書を刊行するにあたり，SCCソフトウエアコンサルタント株式会社，株式会社イーディシー，日本電子開発株式会社各社の技術部には多大の応援をいただきました。厚くお礼申し上げます。

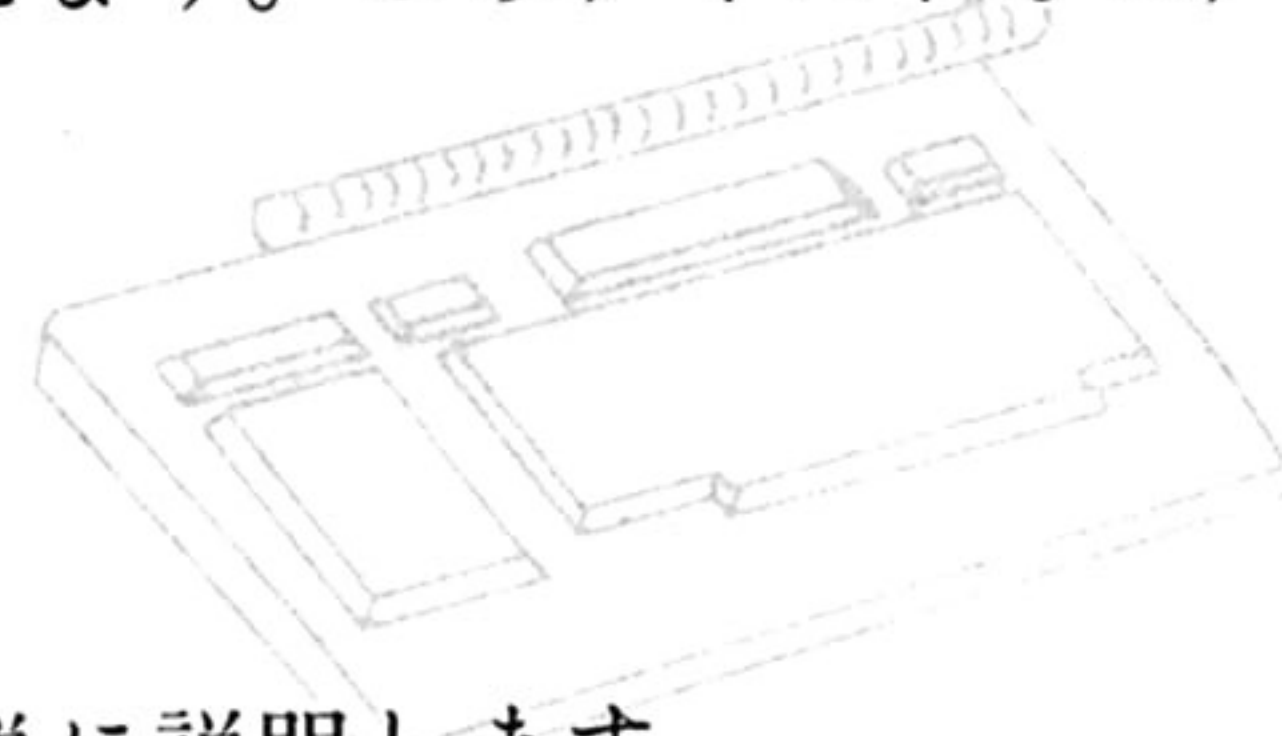
目　　次

1章　プログラムを作る前の基礎知識

パソコンは，テレビやステレオあるいは冷蔵庫などと違って，買ってきて電源を入れればすぐに使えるというわけにはいきません。自動車や楽器と同様に，その使い方をある程度マスター(習得)しないと，本当の性能を発揮させることはできません。

それでは，パソコンに仕事をさせるにはどうしたらよいのでしょうか。それには，まずパソコンが理解できる言葉で仕事の手順をあらかじめ教えておいて，必要に応じて指令を出すことによって仕事をさせます。このパソコンが理解できる仕事の手順の集まりを，特に「プログラム」と呼んでいます。

この章では，そのプログラムを作る前にあらかじめ覚えておいた方がよいような，操作上の知識などを説明します。

1.1　N_{88}DISK-BASICの動かし方

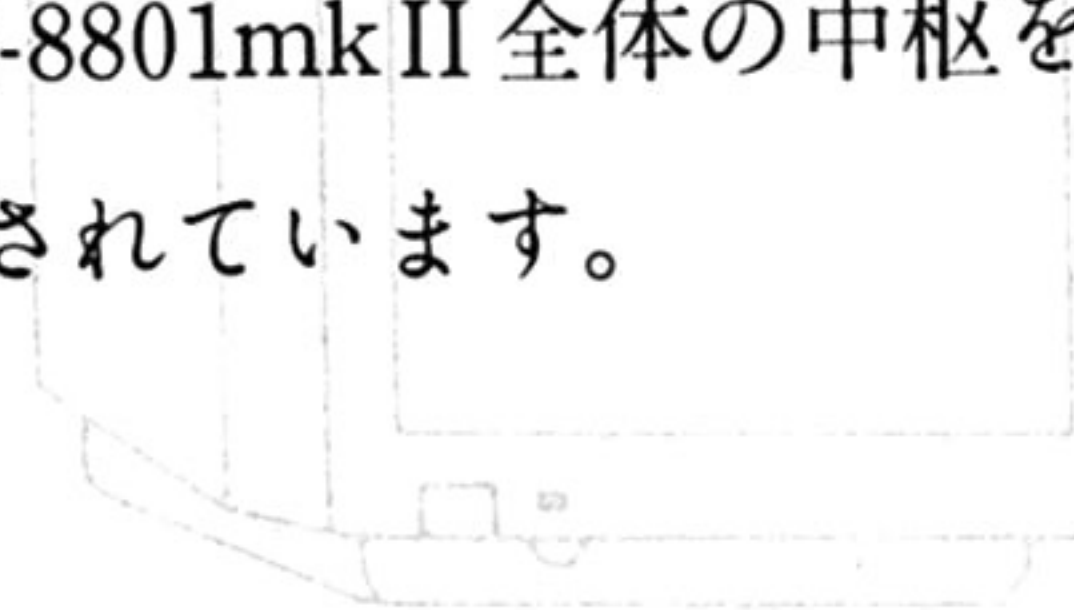
PC-8801mk IIは，N_{88}-BASIC，N_{88}DISK-BASICおよびPC-8000と同じ使い方をするN-BASIC，DISK-BASICの四つのBASICが使えます。このテキストでは，そのうちのN_{88}DISK-BASICを使って学習していきます。

1.1.1　使用する装置のあらまし

このテキストで使用する装置について，簡単に説明します。

(1)　本体

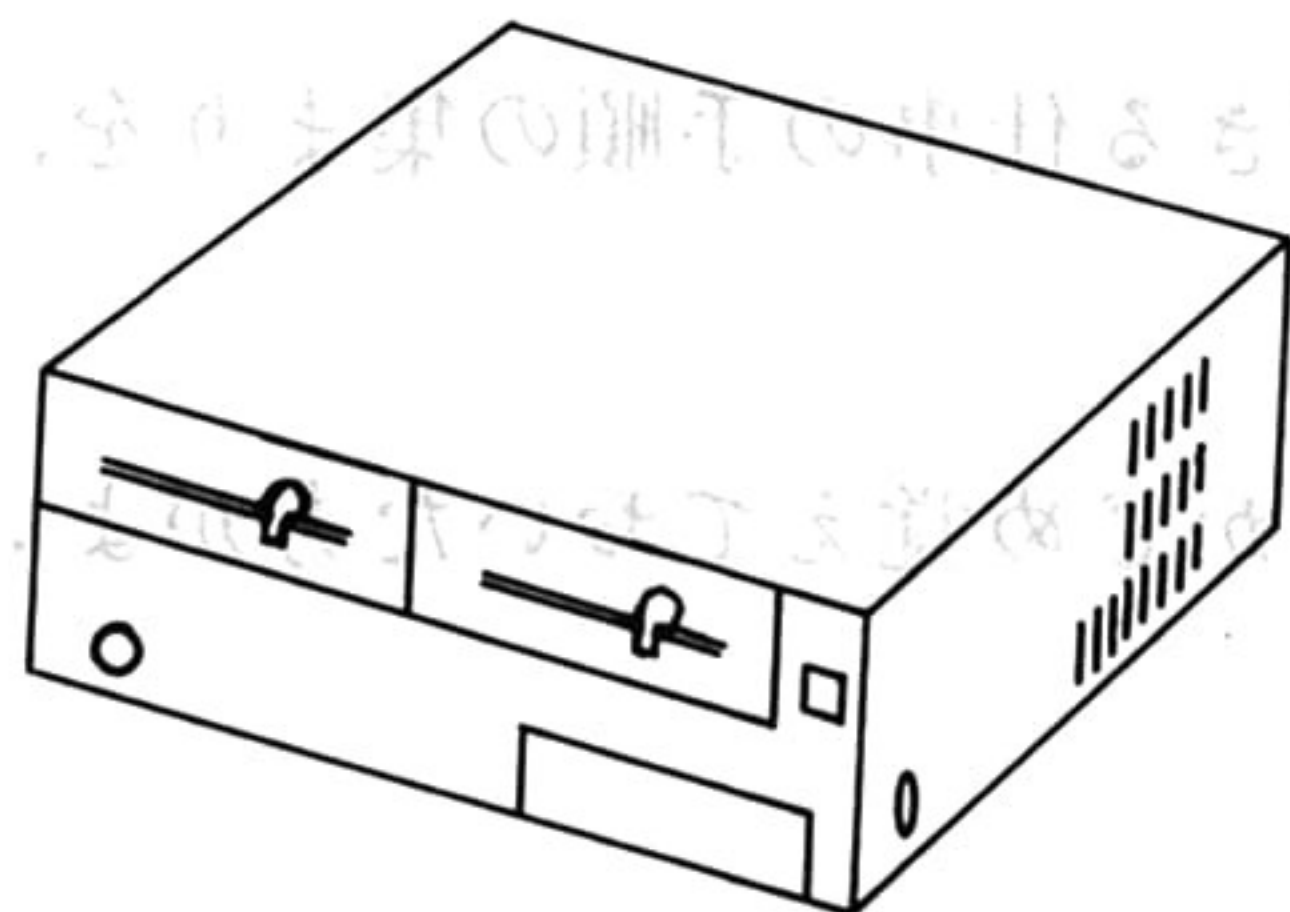
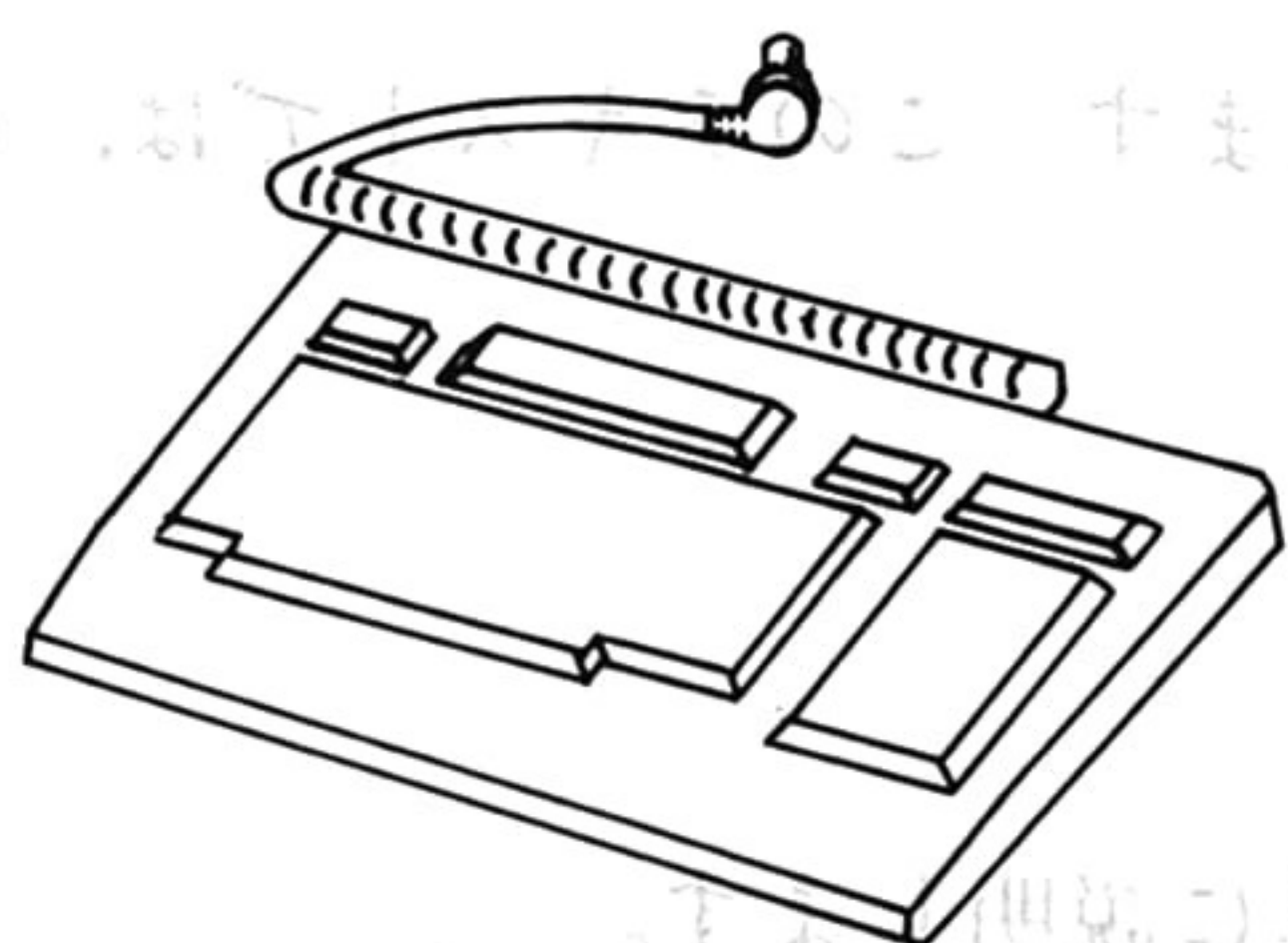
パソコンの本体は人間にたとえると頭脳に相当します。本体はものを考える部分とものを覚える部分に分かれていて，考える部分をCPU (Central Processing Unit)と呼び，覚える部分をメモリー (Memory)と呼びます。

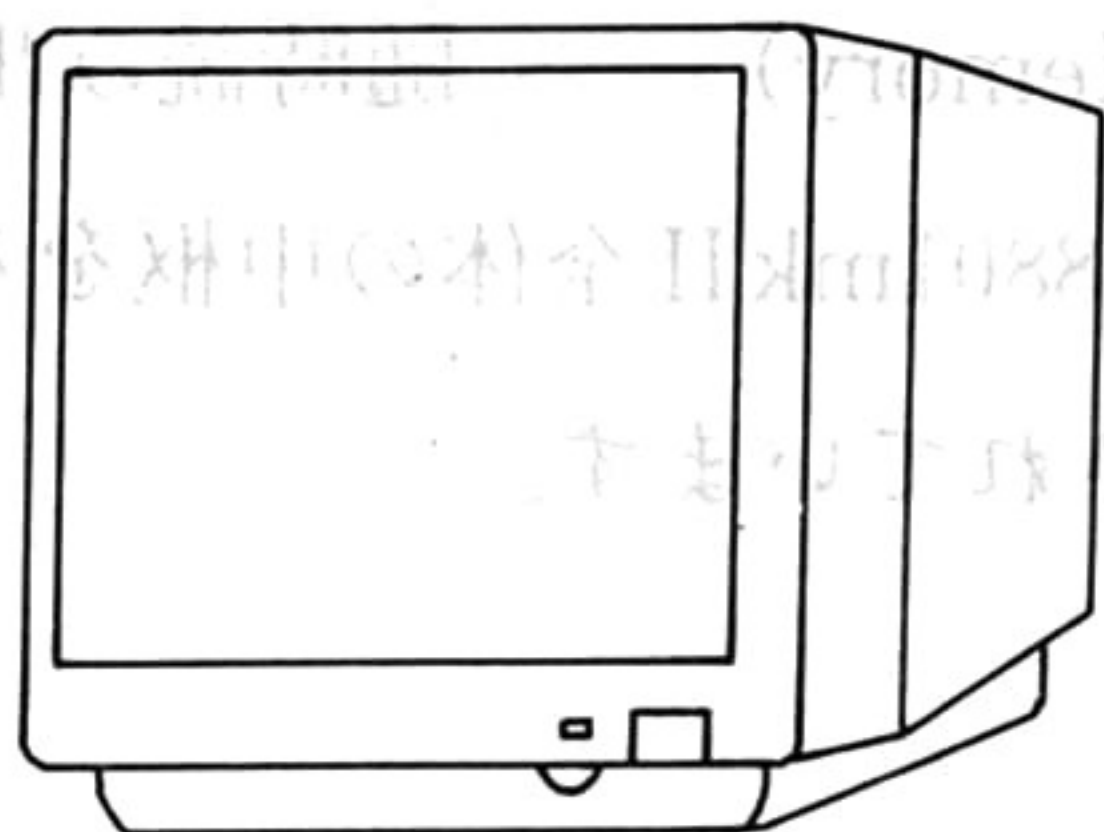
・CPU(中央処理装置) ―― 考える所
・メモリー(記憶装置) ――― 覚える所
　→ROM(Read Only Memory) ―――― 読み出し専用メモリー
　→RAM(Random Access Memory) ―― 随時読み出しメモリー

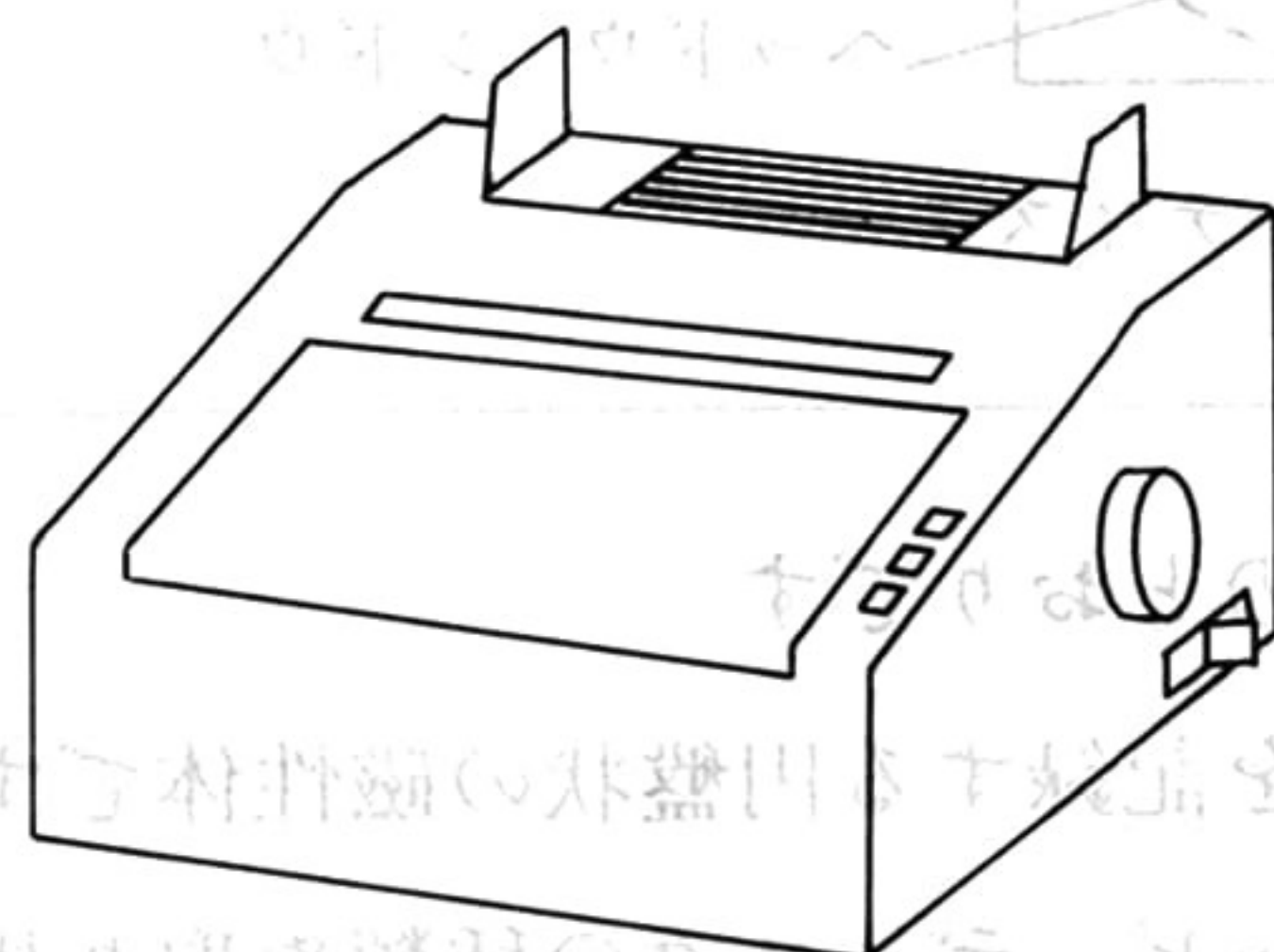
CPUはマイクロプロセッサそのもので，PC-8801mk II全体の中枢をなしています。また，メモリーはROMとRAMの二つの部分に区別されています。

(2)　フロッピーディスク装置

重要なことを忘れないようにするために，人間はメモを取ったりノートに書き留めたりしますが，パソコンはカセットテープやフロッピーディスクシートに書き込みます。カセット

テープに情報を書いたり読み出したりする装置をカセットテープ装置，フロッピーディスクシートに情報を書いたり読み出したりする装置をフロッピーディスク装置と呼びます。なお，カセットテープやフロッピーディスクシートには情報が磁気化されて書かれるため，人間が直接見ても何が書いてあるか判別できません。

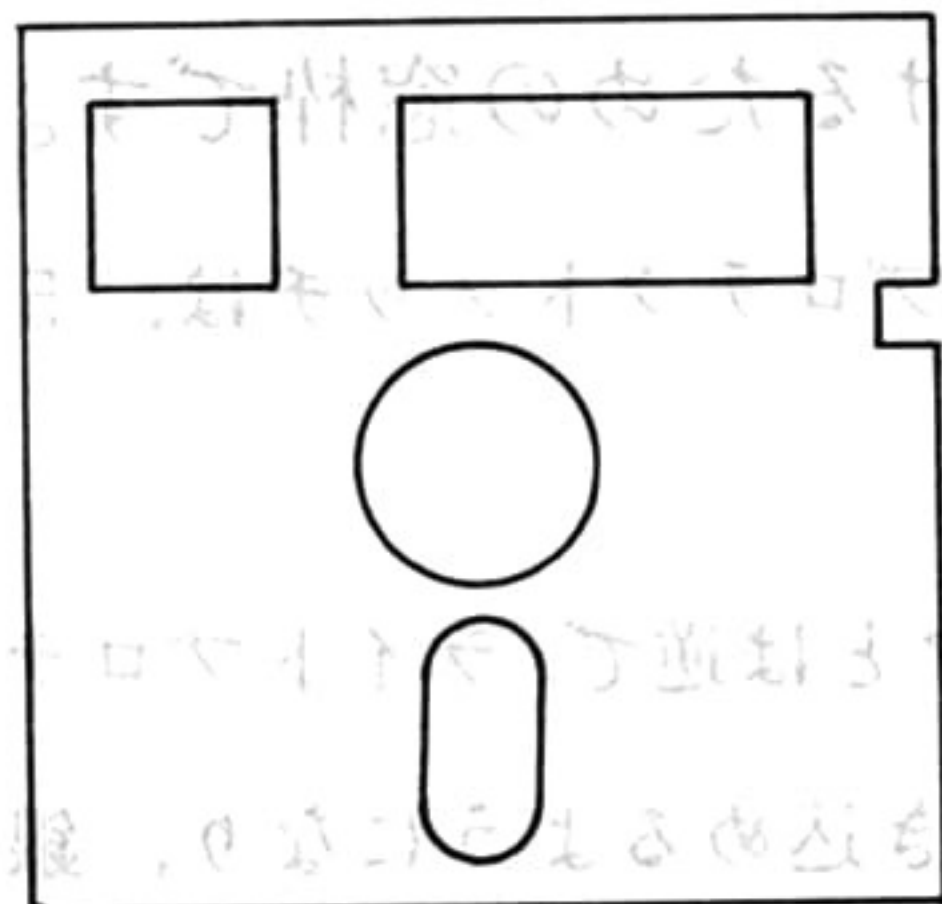
また，PC-8801mkII Model20/Model30では，フロッピーディスク装置が本体に内蔵されています。(このテキストではModel30を使用しています。)

(3) キーボード

人に仕事を頼む場合，自分の考えていることや要求しようとすることを口にしたり文章で表現したりして相手に伝えます。相手はそれを耳できいたり目で読んだりして受け取ります。しかし，PC-8801mkIIには耳や目がないので，PC-8801mkIIに自分の考えや意志を伝えるにはキーボード(Keyboard)を使用します。

(4) CRTディスプレイ装置

人間は自分の考えていることを普通，口から言葉にして相手に伝えます。ところがPC-8801 mk IIには口がないので，代わりにCRTディスプレイ装置(テレビ画面，CRT)を使って人間にいろいろなことを伝えています。

（5） プリンタ装置

CRTに表示された文字や図形は一時的なもので，電源を切ると消えてしまいます。情報を記録としていつまでも保存しておくには，プリンタ(Printer) 装置に印字するようにします。

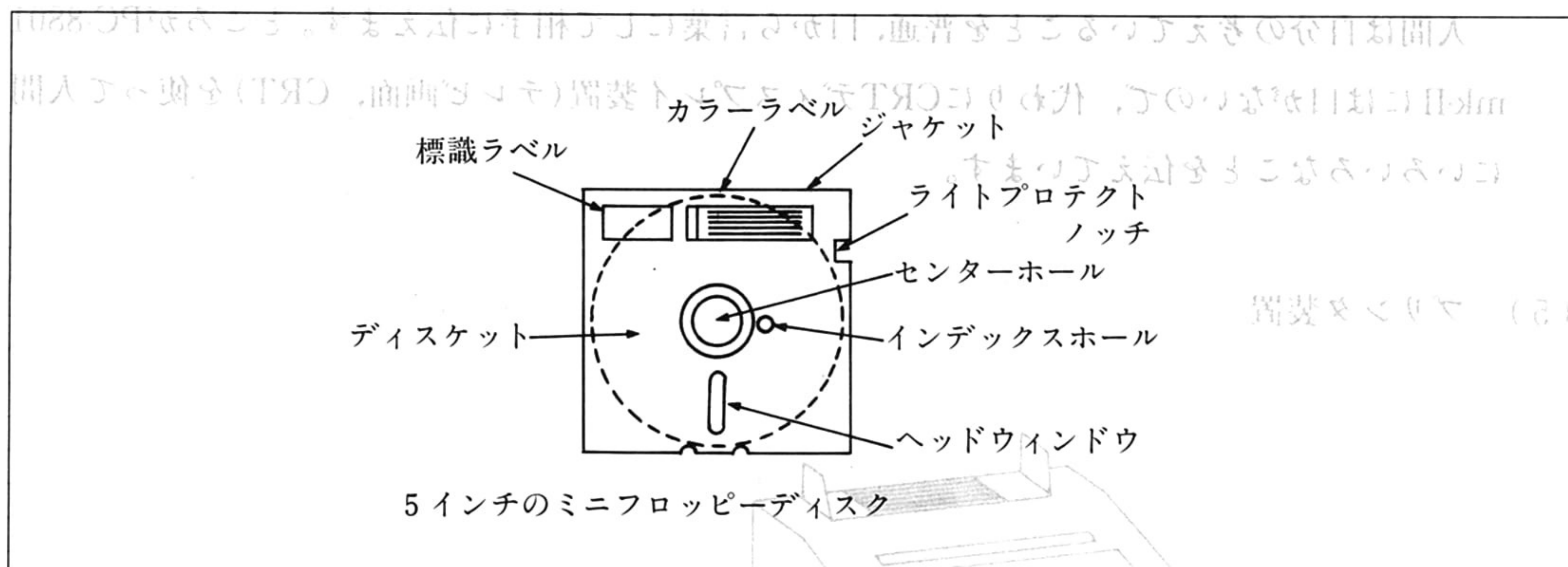
1.1.2 フロッピーディスクについて

（1） フロッピーディスクとは

フロッピーディスクとは録音テープのように情報を磁気化して記録するもので，ちょうどレコード盤をジャケットに入れた時のような形をしています。またフロッピーディスクの円盤上にはレコードの溝に相当する記録部分があって，「トラック」と呼ばれています。一枚のフロッピーディスクにはそのトラックが同心円状に数十本作られていて，データやプログラムの読み書きはトラックに沿って行われます。

（2） フロッピーディスクの各名称

PC-8801mk IIで使用する5インチ・ミニ・フロッピーディスクの構造は，次のようになっています。

標識ラベル
カラーラベル
ジャケット
ライトプロテクト
ノッチ
センターホール
ディスケット
インデックスホール
ヘッドウィンドウ

5インチのミニフロッピーディスク

また，それぞれの名称とその意味は次のとおりです。

・ディスケット……………………情報を記録する円盤状の磁性体です。

・標識ラベル………………………フロッピーディスクの種類や取り扱う方向を示しています。また管理番号や使用開始日などの固定情報の記録にも使います。

・ジョブ用カラーラベル………フロッピーディスクに記録しているファイルやプログラムの名称を記入するために使います。

・ジャケット………………………ディスケットを保護するためのケースです。

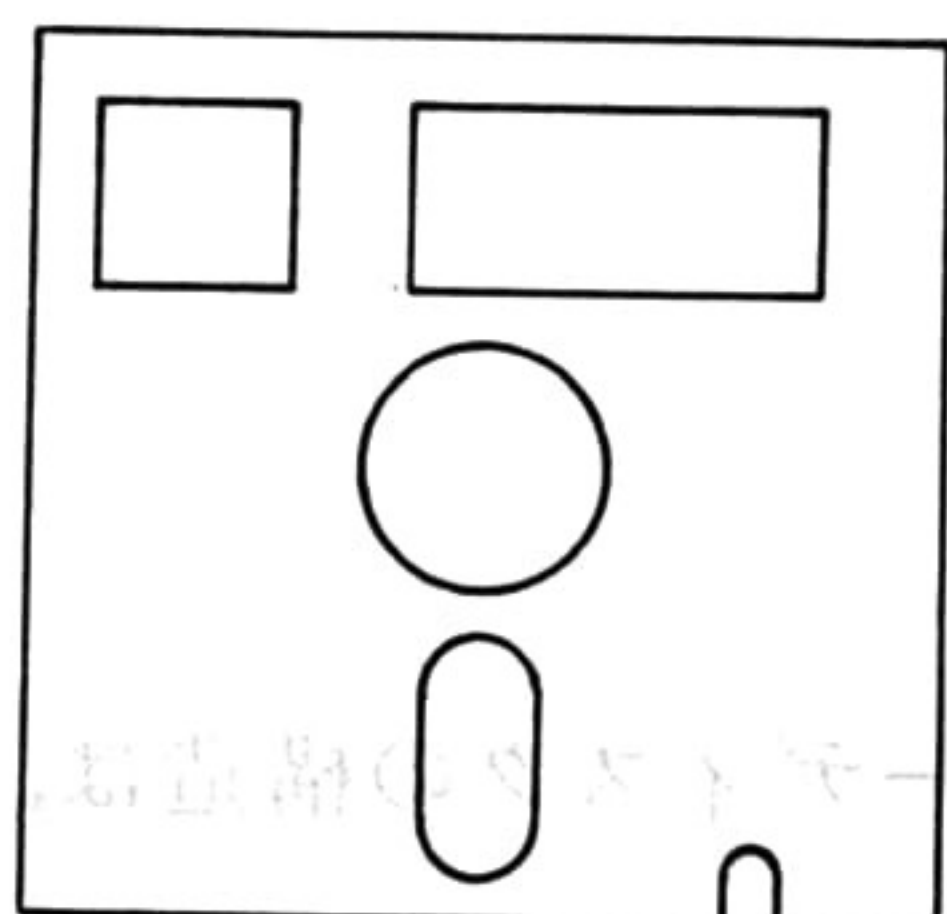
・ライトプロテクトノッチ……ディスケットに記録している情報を保護するかどうかを機械的に判断します。ここに銀紙（ライトプロテクトシール）をはると書き込めなくなります。

・センターホール………………フロッピーディスク装置の回転軸が挿入されるところです。

・インデックスホール…………回転しているディスケット中のどの位置に情報が記録されているかを調べるために起点を与える標識です。

・ヘッドアクセスホール………情報を読み書きするための磁気ヘッドをディスケットの記録面に近づけるための窓枠です。

（注）　8インチの標準フロッピーディスクのライトプロテクトノッチは，5インチのディスクと違ってジャケットの下の方にあります。

また，8インチのディスクは5インチのディスクとは逆で，ライトプロテクトノッチに銀紙をはると情報が書き込めるようになり，銀紙をはがすと書き込めなくなります。

1.1.3　PC-8801mkIIの動かし方

（1）　あらまし

PC-8801mkIIでN_{88}-BASICを使うには二通りの方法があります。一つはそのまま電源を入れてN_{88}-BASICを起動させる方法，もう一つはシステムディスクを使ってN_{88}DISK-BASICを起動させる方法です。

このテキストでは，N_{88}DISK-BASICを使って学習していくことにします。

（2）　N_{88}DISK-BASICの起動手順

N_{88}DISK-BASICの起動は，次の手順で行います。

① 電源の投入

電源

・各装置（本体，CRT，プリンタ）の電源スイッチをONにします。

⇩

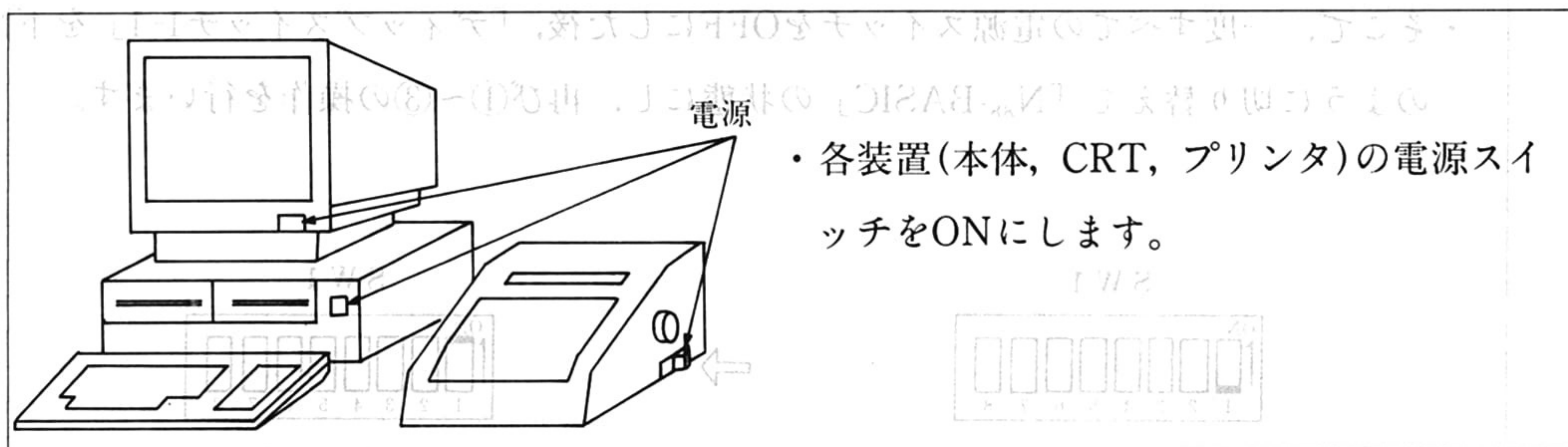
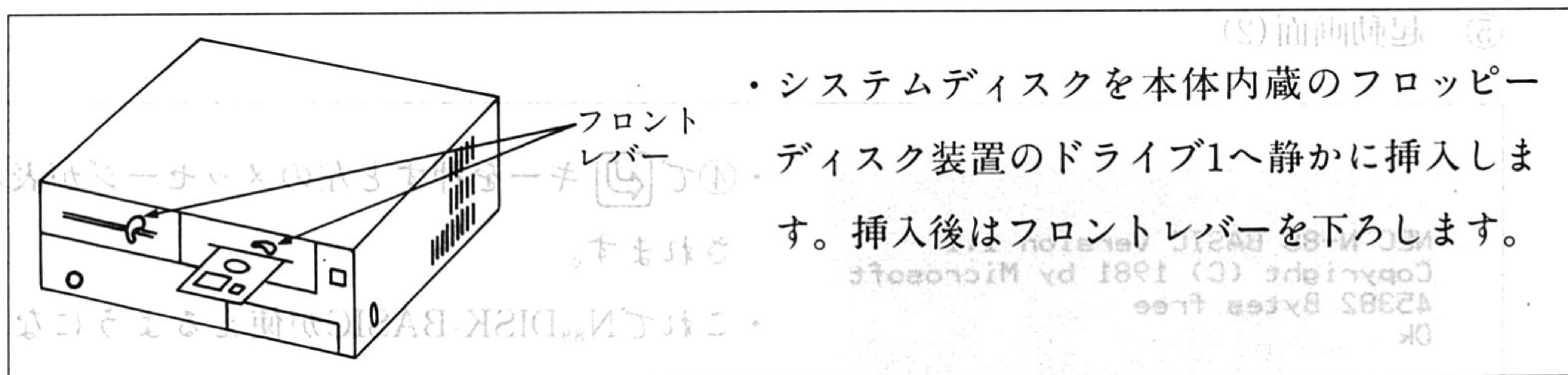
② システムディスクのセット

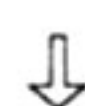
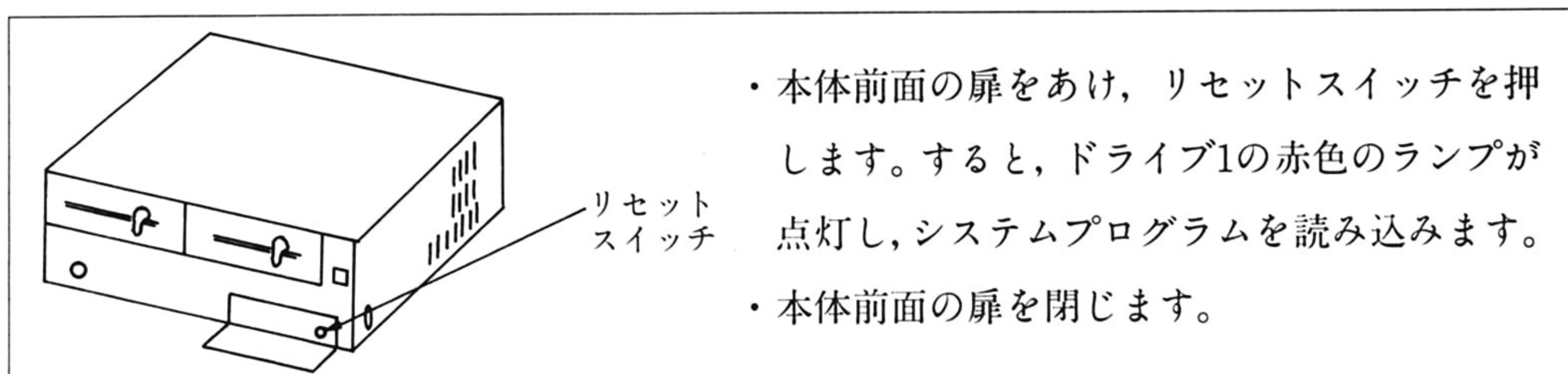
フロント
レバー

・システムディスクを本体内蔵のフロッピーディスク装置のドライブ1へ静かに挿入します。挿入後はフロントレバーを下ろします。

⇩

③ リセットスイッチを押す

リセット
スイッチ

・本体前面の扉をあけ，リセットスイッチを押します。すると，ドライブ1の赤色のランプが点灯し，システムプログラムを読み込みます。
・本体前面の扉を閉じます。

⇩

④　起動画面(1)

```
Disk version [Aug 19,1983]
How many files(0-15)?
```

・ドライブ1の赤色のランプが消えると，左のメッセージが画面に表示されます。このときは，⏎ キーを押します。

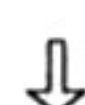
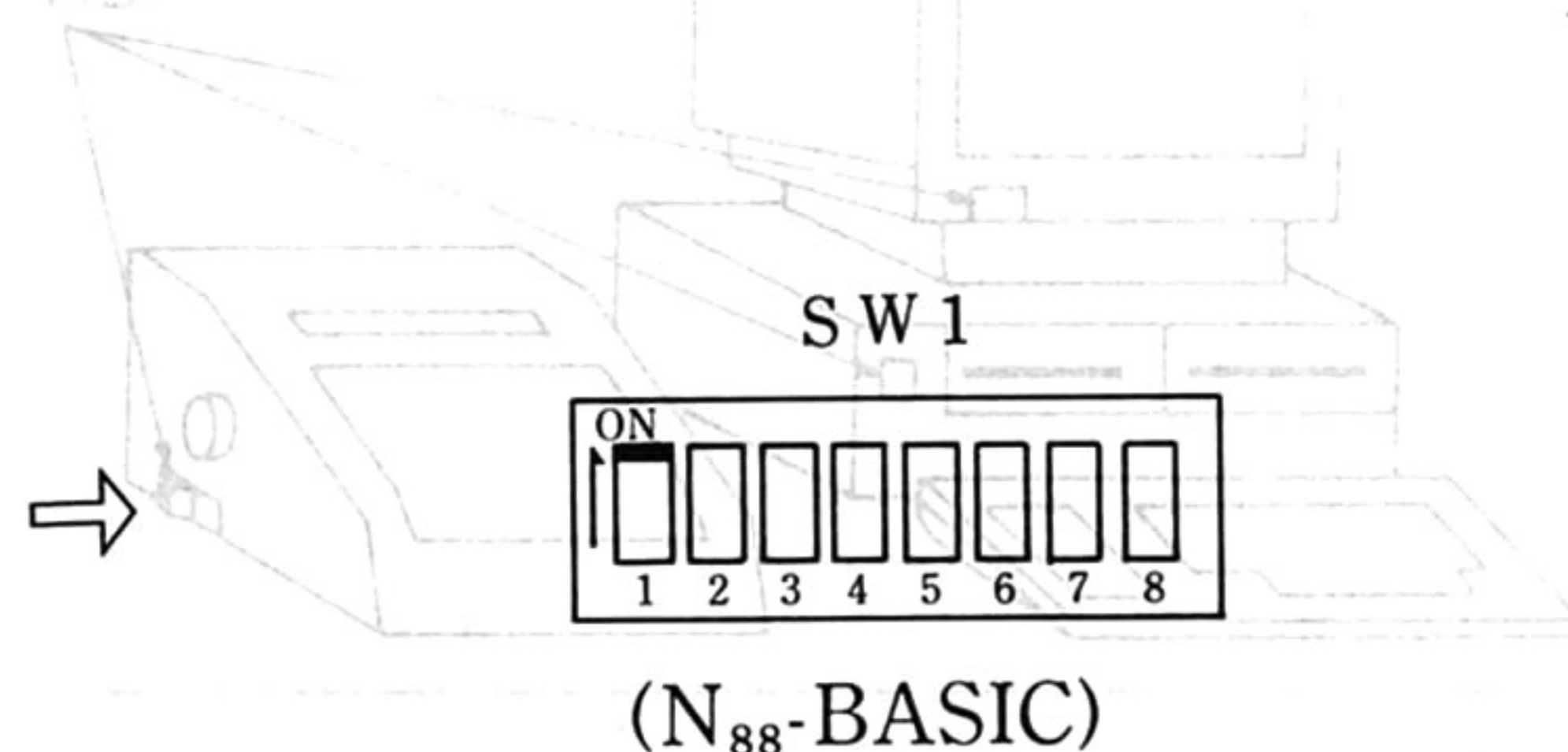
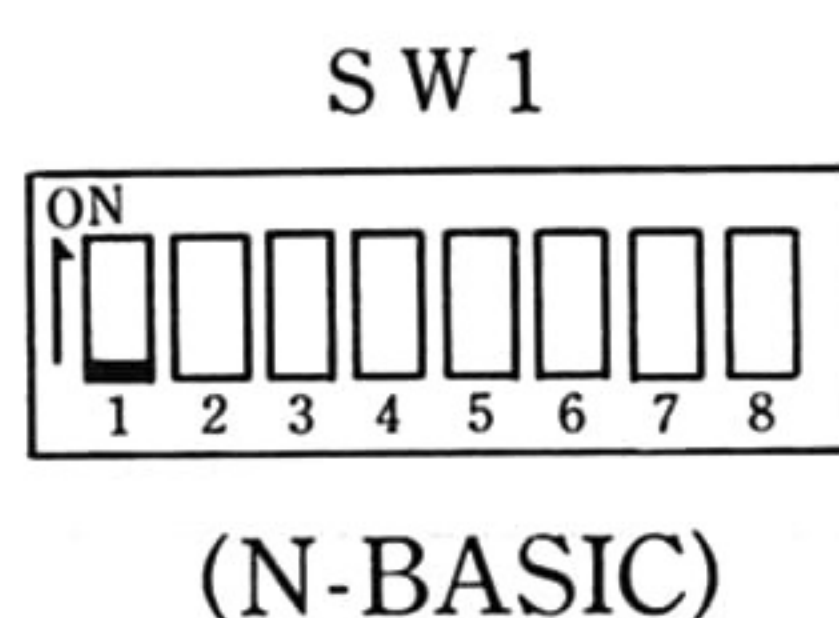
・「How many files(0-15)？」については，下巻で学習します。

・①〜③の操作を行った後，ドライブ1の赤ランプの点滅を繰り返しいつまでも左のメッセージが表示されないことがあります。

・これは，本体前面の扉の内側にある「ディップスイッチ」と呼ばれるものが「N-BASIC」の状態になっているためです。

・そこで，一度すべての電源スイッチをOFFにした後，「ディップスイッチ1-1」を下図のように切り替えて「N_{88}-BASIC」の状態にし，再び①〜③の操作を行います。

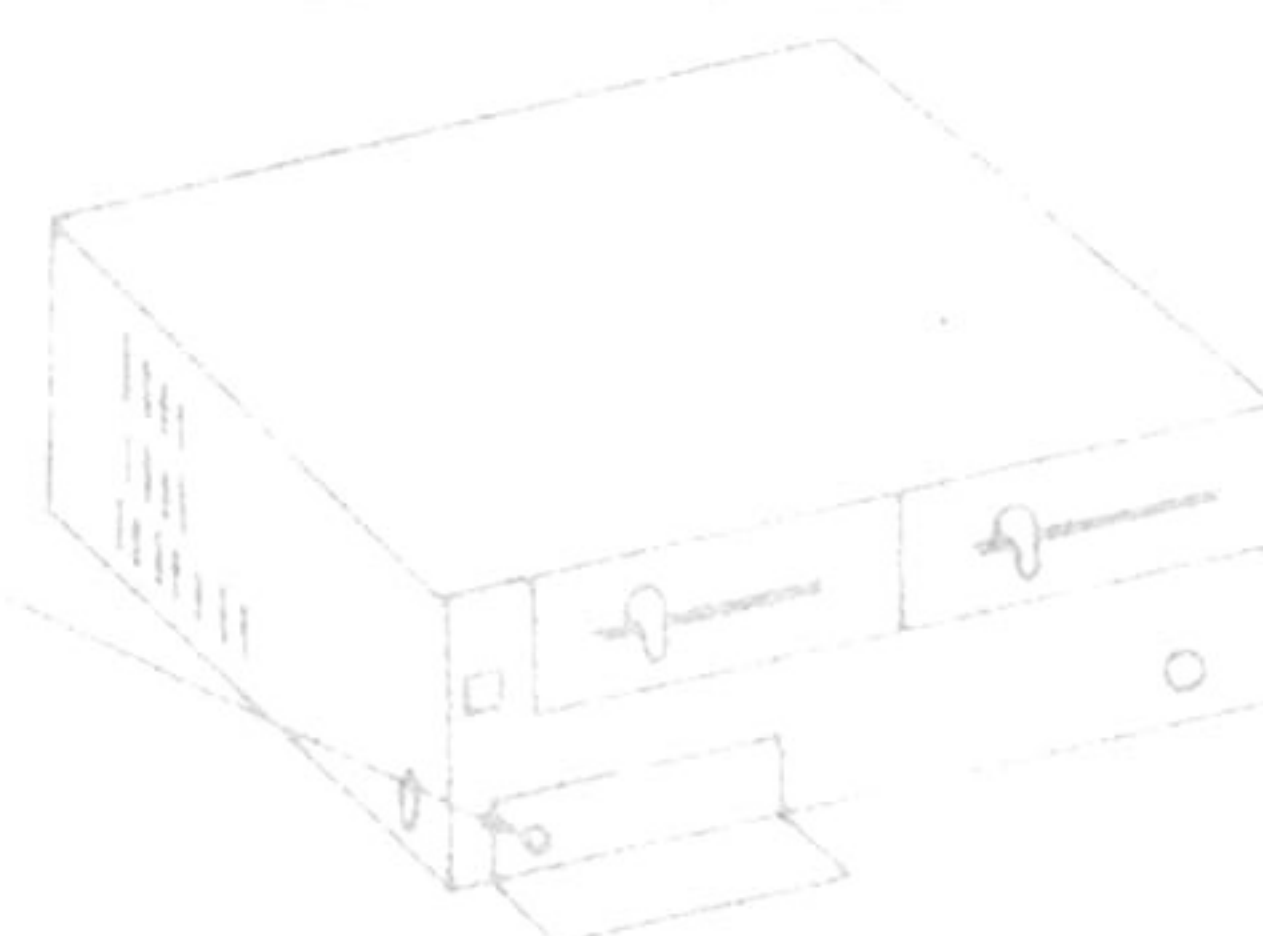
SW1　ON　1 2 3 4 5 6 7 8　(N-BASIC)　⇨　SW1　ON　1 2 3 4 5 6 7 8　(N_{88}-BASIC)

⇩

⑤　起動画面(2)

```
NEC N-88 BASIC Version 1.1
Copyright (C) 1981 by Microsoft
45382 Bytes free
Ok
```

・④で ⏎ キーを押すと左のメッセージが表示されます。

・これでN_{88}DISK-BASICが使えるようになります。

〈一口メモ〉

フロッピーディスクを取り扱うときの注意

フロッピーディスクはとても薄く作られている磁性体です。そのために取り扱いには次のような注意が必要です。

① フロッピーディスクのディスケット部分（磁性体の部分）を直接手などで触ってはいけません。直接触ると指紋などが付き，ゴミの付着を招いてフロッピーディスクやディスク装置の損傷の原因にもなります。

② 直接日光や高温（65℃以上）にさらしてはいけません。また，水などで濡らしてもいけません。

③ フロッピーディスクを折り曲げてはいけません。

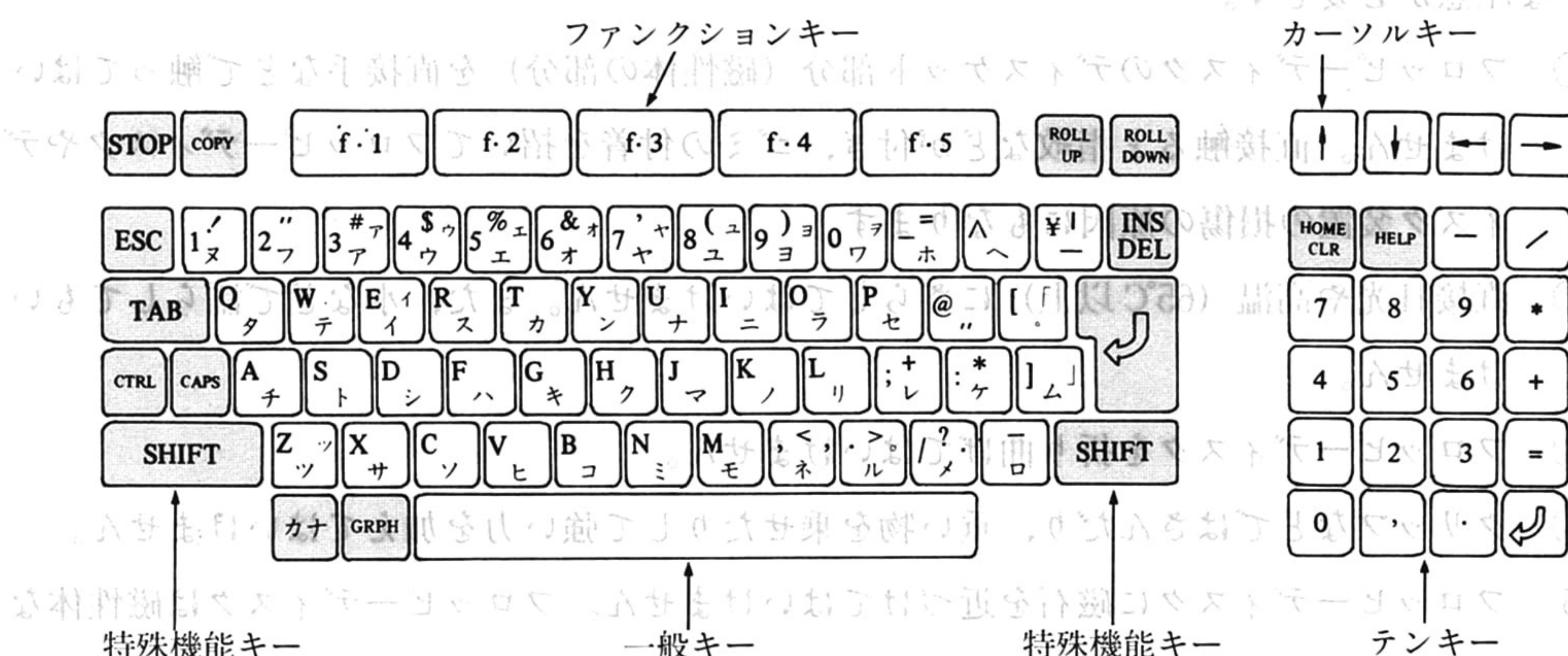
④ クリップなどではさんだり，重い物を乗せたりして強い力を加えてはいけません。

⑤ フロッピーディスクに磁石を近づけてはいけません。フロッピーディスクは磁性体なので磁石の磁気でデータやプログラムが壊される恐れがあります。

1.2　キーボードの使い方

キーボードを上からみると次のようになっています。

ファンクションキー　カーソルキー

特殊機能キー　一般キー　特殊機能キー　テンキー

キーボード上のキーにはそれぞれ異なった働きがありますが，ここではそれらの機能について説明します。

1.2.1　英字(アルファベット)の表示

英字を表示するには，英字が書かれているキーを使います。

(1)　英小文字

英字の小文字を表示するには，英字のキーをそのまま押します。また，CAPS キーが押されているときは，SHIFT キーを押しながら英字のキーを押します。

・CAPS キーが出ている状態のとき

A チ キーを5回押す ――――→ a a a a a

・CAPS キーが引っ込んでいる状態のとき

SHIFT ⊕ B コ キーを5回押す ――――→ b b b b b

(注)　「⊕」は「左側のキーを押しながら右側のキーを押す」という意味です。

（2） 英大文字

英字の大文字を表示するには，SHIFTキーを押しながら英字キーを押します。また，CAPSキーが押されているときは，そのまま英字キーを押します。

・CAPSキーが出ている状態のとき

SHIFT ⊕ A チ キーを5回押す ――→ AAAAA

・CAPSキーが引っ込んでいる状態のとき

B コ キーを5回押す ――→ BBBBB

1.2.2 数字の表示

数字を表示するには，数字の書かれているキーを使います。数字のキーは一般キーとテンキーの両方に用意されています。

・一般キーの場合

1 ! ヌ キーを5回押す ――→ 11111

・テンキーの場合

2 キーを5回押す ――→ 22222

数字のキーは，CAPSキーが押されていてもこれには関係なく数字を表示します。

〈一口メモ〉

カーソル

カーソル(Cursor)とは画面上で点滅している「■」のマークのことで，キーインされた文字が画面上のどの位置に表示されるかを示します。また，このカーソルだけを上下左右に移動させて指標として使うこともできます。

1.2.3　カナ文字の表示

カナ文字を表示するには，カナ文字が書かれているキーを使います。

（1）　カナ大文字

カナ文字の大文字を表示するには，[カナ]キーを押してロック(引っ込んでいる状態)させた後カナ文字が書かれているキーを押します。

([カナ]キーをロックした後)

[3ア]キーを5回押す ──────→ アアアアア

（2）　カナ小文字

カナ文字の小文字を表示するには，[カナ]キーをロックさせた後[SHIFT]キーを押しながらカナ文字の小文字が書かれているキーを押します。

([カナ]キーをロックした後)

[SHIFT] ⊕ [Eイ] キーを5回押す ──────→ ィィィィィ

1.2.4　特殊記号の表示

「”」や「<」などの特殊記号を表示するには，特殊記号が書かれているキーを使います。

（1）[SHIFT]キーを併用する文字

特殊記号の大半は[SHIFT]キーを押しながらそのキーを押して表示します。

・[SHIFT] ⊕ [5%エ] キーを5回押す ──────→ %%%%%

また，[SHIFT]キーを併用して表示する特殊記号とその読み方を次に示します。

記　号	読 み 方	記　号	読 み 方
!	エクスクラメーション	)	右カッコ
”	クォーテーション	=	イコール
#	ナンバーサイン(イゲタ)	+	プラス
$	ダラーサイン	*	アスタリスク
%	パーセント	<	不等号(小)
&	アンパーサンド	>	不等号(大)
’	アポストロフィー	?	クエスションマーク
(	左カッコ		

ただし，テンキー側にある「=」，「+」，「*」は[SHIFT]，[CAPS]，[カナ]に関係なく利用することができます。

(2) SHIFT キーを併用しない文字

特殊記号の一部は SHIFT キーを押さずにそのキーを押して表示します。

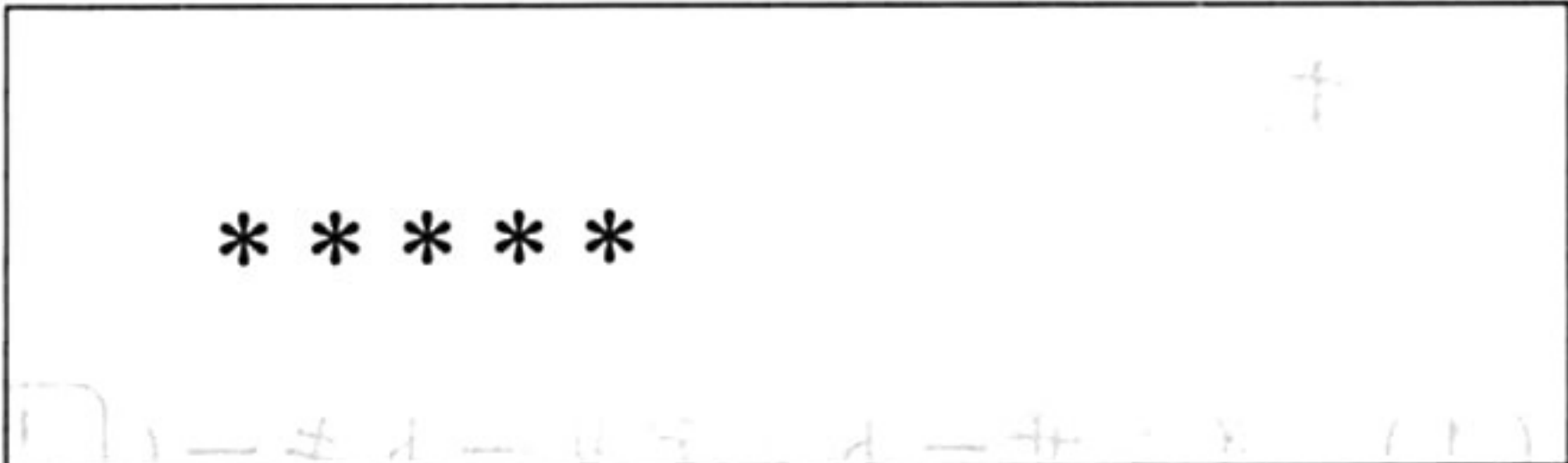
・ ¥ キーを5回押す ――――→ ¥¥¥¥¥

また，SHIFT キーを併用しないで表示する特殊記号とその読み方を次に示します。

記　号	読 み 方	記　号	読 み 方
―	マイナス，ハイフォン	;	セミコロン
^	アップアロー(ヤマガタ)	:	コロン
¥	エンサイン	,	コンマ(カンマ)
@	アットマーク	.	ピリオド
[	左角カッコ	／	スラッシュ
]	右角カッコ		

(3) テンキーを使う文字

演算に使われるような特殊記号はテンキーにも用意されています。

・ * キーを5回押す ――――→ *****

また，テンキーに用意されている特殊記号は，次の7種類です。

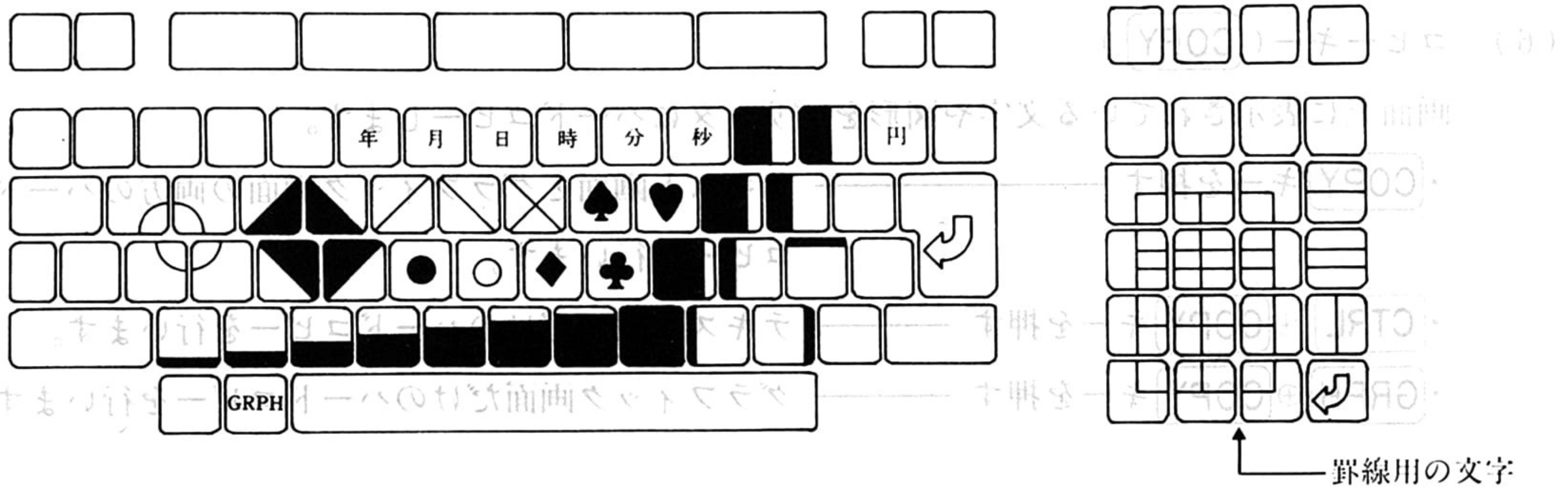
－，／，*，＋，＝，．，，

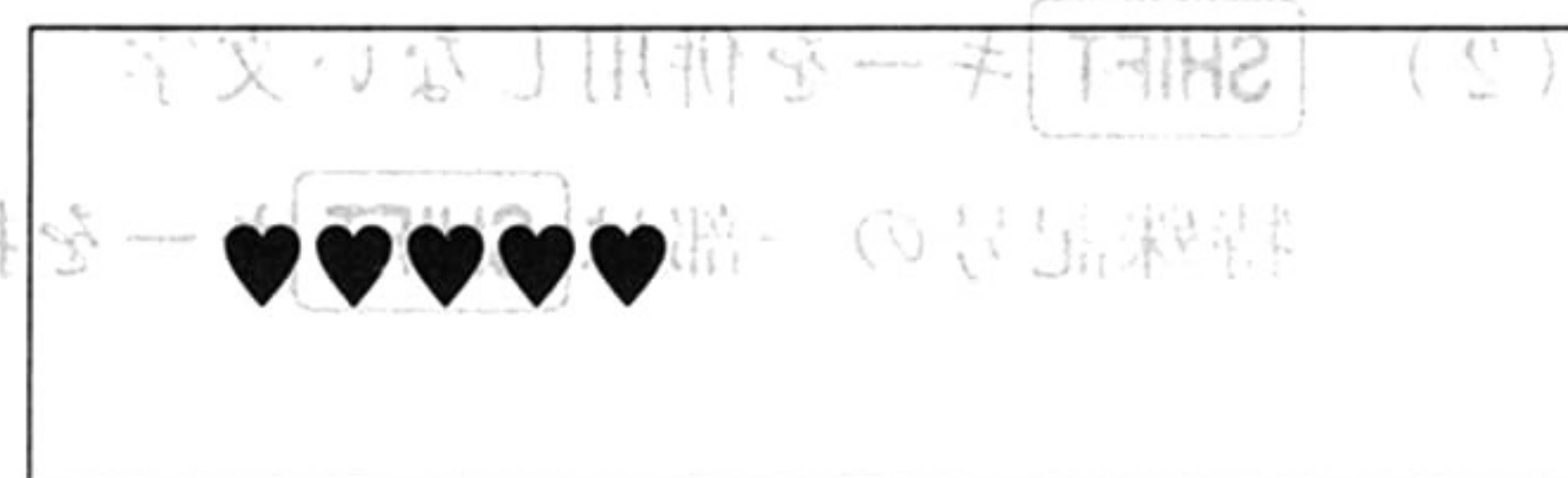
1.2.5 グラフィック文字の表示

グラフィック文字を表示するには，GRPH キーを併用します。GRPH キーを押したとき，キーボードの配列は次のように変わります。

年　月　日　時　分　秒　円

GRPH

罫線用の文字

GRPH キーを押したときのキー配列

・GRPH ⊕ ｡ｦ キーを5回押す ――→ ♥♥♥♥♥

1.2.6 特殊な機能を持つキー

そのほか，キーボードには特殊な働きをするキーが用意されています。

（1） カーソル移動キー（↑ ↓ ← →）

画面上のカーソル（点滅している一文字分の箱で，次に文字を表示する位置を表しています）を上下左右に動かします。

（2） リターンキー（↵）

プログラムやデータの入力のときに使います。一行入力した後このキーを押すとその行の入力が終わり，カーソルが次の行の先頭に移動します。

（3） ホーム・クリアキー（HOME CLR）

そのまま押すと，画面上の文字を消した後画面の左上隅にカーソルを移動させます。

またSHIFTキーを併用すると，画面上の文字は消さずにカーソルだけが左上隅に移動します。

（4） インサート／デリートキー（INS DEL）

そのまま押すと，カーソルの左側の1文字を削除し後ろの文字を前へ詰めます。

またSHIFTキーを併用すると，文字を挿入できる状態になり，それ以降に押した文字が挿入されます。挿入の状態はもう一度特殊機能キー（INS DEL キーを含む）が押されるまで続きます。

（5） タブキー（TAB）

8桁ごとにカーソルの移動を行います。

（6） コピーキー（COPY）

画面上に表示されている文字や図形をプリンタにハードコピーします。

・COPYキーを押す ―――― テキスト画面とグラフィック画面の両方のハードコピーを行います。

・CTRL ⊕COPYキーを押す ―――― テキスト画面だけのハードコピーを行います。

・GRPH ⊕COPYキーを押す ―――― グラフィック画面だけのハードコピーを行います。

（7） ストップキー（STOP）

プログラムなどの実行を中断します。

（８） エスケープキー（ESC）

N-BASICモードで，プログラムの表示や実行を一時的に停止させるときに使います。任意のキーを押すと，実行を再開します。

（９） ヘルプキー（HELP）

プログラムの実行中にエラーが起きたとき，通常はエラーメッセージが表示されますが，このときにエラーのおきた部分の命令を表示させることができます。

(10) ロールアップ／ロールダウンキー（ROLL UP ROLL DOWN）

画面上の文字を上下させます。

〈一ロメモ〉

テキスト画面とグラフィック画面

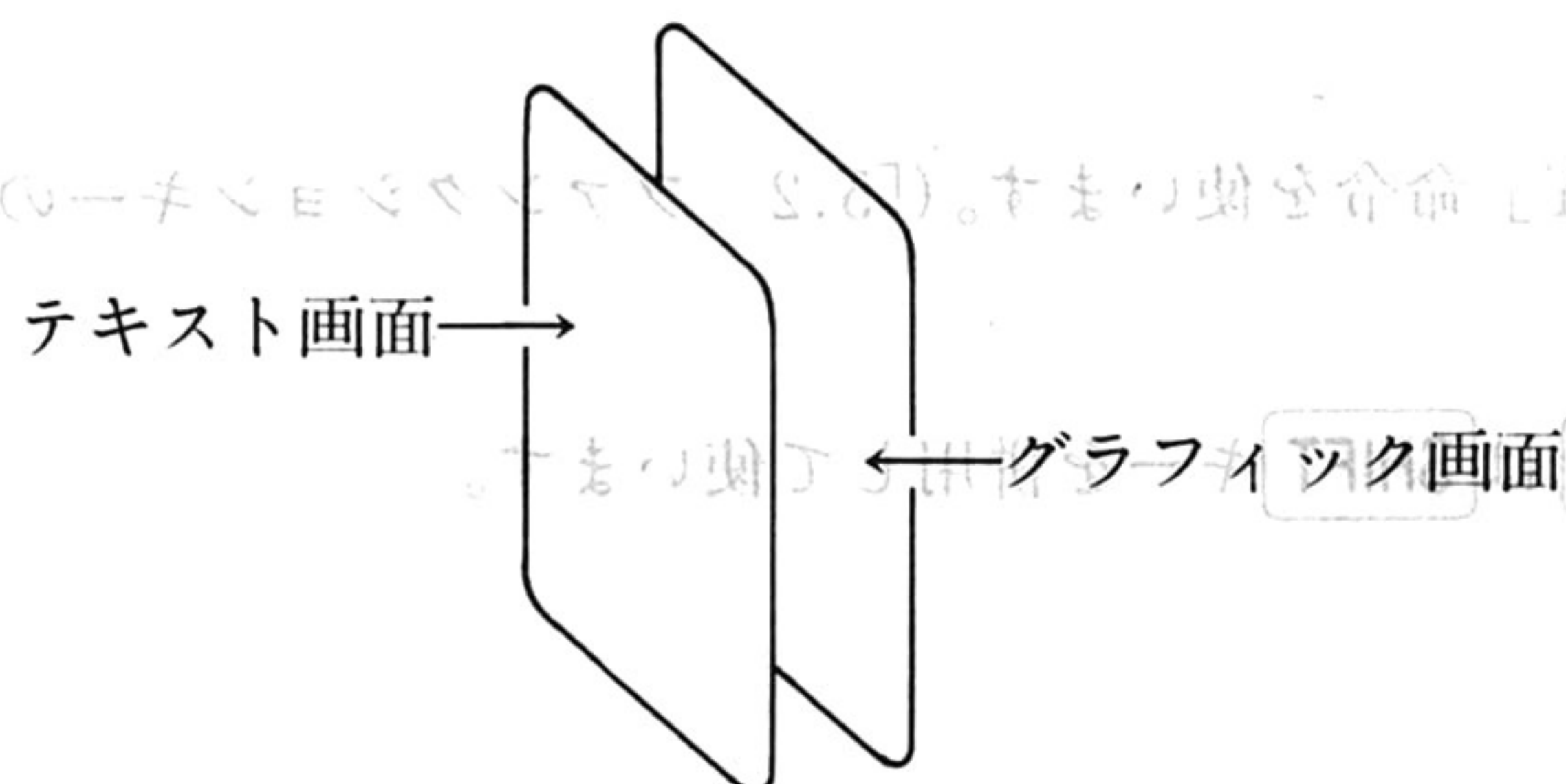
CRTディスプレイの画面は「テキスト画面」と「グラフィック画面」に大別でき，現在大半のパソコンがこの二つの画面を持っています。

テキスト画面は，キーボードから入力した文字やプログラムのリスト，数値などを表示する画面です。

グラフィック画面は，絵や図形などを表示するための専用画面です。漢字もこの画面に表示します。

また，テキスト画面とグラフィック画面は，それぞれ独立していてスライドを二枚重ねたような形で設定されています。二つの画面のイメージを下の図で示します。

テキスト画面→

←グラフィック画面

(11)　コントロールキー(CTRL)

他のキーと併用して特殊な機能が使えます。組み合わせと機能は次のとおりです。

組み合わせ	機　能
CTRL ⊕ A	HELPキーと同じ働きをします。
CTRL ⊕ B	カーソルを一項目ごとに左へ移動させます。
CTRL ⊕ C	STOPキーと同じ働きをします。
CTRL ⊕ D	カーソルが点滅している項目を削除します。
CTRL ⊕ E	カーソルが点滅している位置からその行の最後の文字までを消します。
CTRL ⊕ F	カーソルを一項目ごとに右へ移動させます。
CTRL ⊕ G	ブザーを鳴らします。
CTRL ⊕ H	INS DEL キーと同じ働きをします。
CTRL ⊕ I	TAB キーと同じ働きをします。
CTRL ⊕ J	挿入モード(SHIFT ⊕ INS DEL キーを押した後)のとき，カーソルの点滅している位置から後の文字が次の行へ移ります。
CTRL ⊕ K	SHIFT ⊕ HOME CLR キーと同じ働きをします。
CTRL ⊕ L	HOME CLRキーと同じ働きをします。
CTRL ⊕ M	⏎ キーと同じ働きをします。
CTRL ⊕ O	プログラムの表示中や実行中に押すと，再びCTRL ⊕ O を押すか実行が終了するまで，画面へ文字を表示しなくなります。
CTRL ⊕ R	SHIFT ⊕ INS DEL キーと同じ働きをします。
CTRL ⊕ S	プログラムの表示や実行を一時的に停止します。任意のキーを押すと，実行を再開します。
CTRL ⊕ U	カーソルが点滅している行を一行すべて消した後，カーソルをその行の左端に移します。
CTRL ⊕ X	カーソルが点滅している行の最後へカーソルを移します。

(12)　ファンクションキー(f·1 ～ f·10)

定義した文字列を取り出すときに使います。文字列は一つのファンクションキーに15文字まで定義できます。

文字列の定義には「KEY」命令を使います。(「5.2　ファンクションキーの利用」で学習します)

なお，f·6 ～ f·10 はSHIFTキーを併用して使います。

1.3　システムディスクの内容を見るには

1.3.1　あらまし

現在，PC-8801mkIIのドライブ1にはシステムディスクがセットされています。このシステムディスクにはN_{88}DISK-BASICだけが記録されているのではなく，デモ・プログラムやユーティリティ・プログラムも記録されています。

それでは，システムディスクの内容を見てみることにしましょう。

1.3.2　システムディスクの内容を画面に表示する

システムディスクには，いろいろなプログラムやデータファイルが記録されています。そのプログラムやデータファイルの名前を画面に表示するために，「FILES」というコマンドが用意されています。

（1）　FILESコマンドとLFILESコマンド

FILESコマンドは，フロッピーディスクの「ファイル管理領域(ディレクトリ)」と呼ばれる部分に記録されている情報(プログラムの名前や大きさ等)を画面に表示するものです。

FILESコマンドの一般形式と書き方例を次に示します。

(FILESコマンドの説明)

＜一般形式＞

FILES　ドライブ番号

・ドライブ番号で指定したドライブに入っているフロッピーディスクのディレクトリリストを画面に表示します。

・ドライブ番号を省略すると「1」を指定したときと同じ結果になります。

＜書き方例＞

① FILES　　ドライブ1に入っているフロッピーディスクのディレクトリリストを画面に表示します。

② FILES　2　　ドライブ2に入っているフロッピーディスクのディレクトリリストを画面に表示します。

また，ディレクトリの内容をプリンタに印字する「LFILES」コマンドもあります。LFILESコマンドの一般形式と書き方例は次のとおりです。

(LFILESコマンドの説明)

<一般形式>

LFILES　ドライブ番号

・ドライブ番号で指定したドライブに入っているフロッピーディスクのディレクトリリストをプリンタに印字します。

・ドライブ番号を省略すると「1」を指定したときと同じ結果になります。

<書き方例>

① LFILES　　ドライブ1に入っているフロッピーディスクのディレクトリリストをプリンタに印字します。

② LFILES 2　ドライブ2に入っているフロッピーディスクのディレクトリリストをプリンタに印字します。

（2） FILESコマンドの実行

それでは，実際にFILESコマンドを使ってみましょう。

設問 1　ドライブ1のフロッピーディスクのディレクトリリストを画面に表示して下さい。

【 実行結果例 】

```
FILES
demo   .n88 7     tutor  .n88 12
dskut1.n88 4     dskut2.n88 9
ngen   .n88 13    @load  .n88 1
@exst *     2     demoex.n88 3
*      *     1     *demo0*     4
*demo1*     1     *demo2*     1
*tutor*     1     $pict0 dat 4
$pict1 dat 4     $pict2 dat 6
$pict3 dat 6     $pict4 dat 10
$pict5 dat 12    $pict6 dat 12
$pict7 dat 14    $pict8 dat 14
Ok
```

【 実行後の説明 】

① キーボードから「FILES」または「files」と入力した後，⏎ キーを押します。

② 「.」や「*」や空白などを含む10文字の文字列が，プログラムやデータファイルの名前です。

③ プログラムまたはデータファイルの名前に続く数字が，そのファイルの大きさを示してます。

〈一口メモ〉

大文字の命令と小文字の命令

NECのPCシリーズのパソコンでは，命令を小文字でキーインしても大文字の命令に自動的に変換してから実行しています。そのため，PC-8801mkIIでは命令を大文字でキーインしても小文字でキーインしても，あるいは大文字と小文字を混合してキーインしても，命令として実行できるわけです。

1.4　ディスクを使う前の基礎作業

実際にプログラムを作っていくと，そのプログラムの保存やデータファイルを作成するために作業用のフロッピーディスク(データディスク)が必要になってきます。また，システムディスクやデータディスクなどを誤って壊してもやり直しができるようにするため，フロッピーディスクの内容を別のフロッピーディスクにコピーすることも必要です。

ここでは，データディスクを作る方法とフロッピーディスクをコピーする方法を学習します。(ここでの学習は，PC-8801mkII Model30を対象にしています)

1.4.1　データ用のディスクを作る方法

(1)　あらまし

PC-8801mkII本体(Model20/Model30)には，システムディスクは添付されていてもデータ作成用のディスク(データディスク)は付いていません。そのため市販されているミニ・フロッピーディスクを買ってきて使用するわけですが，市販のディスクはそのままではPC-8801mkIIで使用できません。市販のディスクをPC-8801mkIIで使えるようにするためには，「ディスクのフォーマッティング」という作業が必要になります。

ディスクのフォーマッティング作業とは，新しいフロッピーディスクや再利用したいフロッピーディスクにトラック番号やセクタ番号(これらは下巻で学習します)を書き込み，ファイル管理領域と呼ばれるフロッピーディスクの使用状況を管理する場所などを初期化することをいいます。

(2)　フォーマッティング

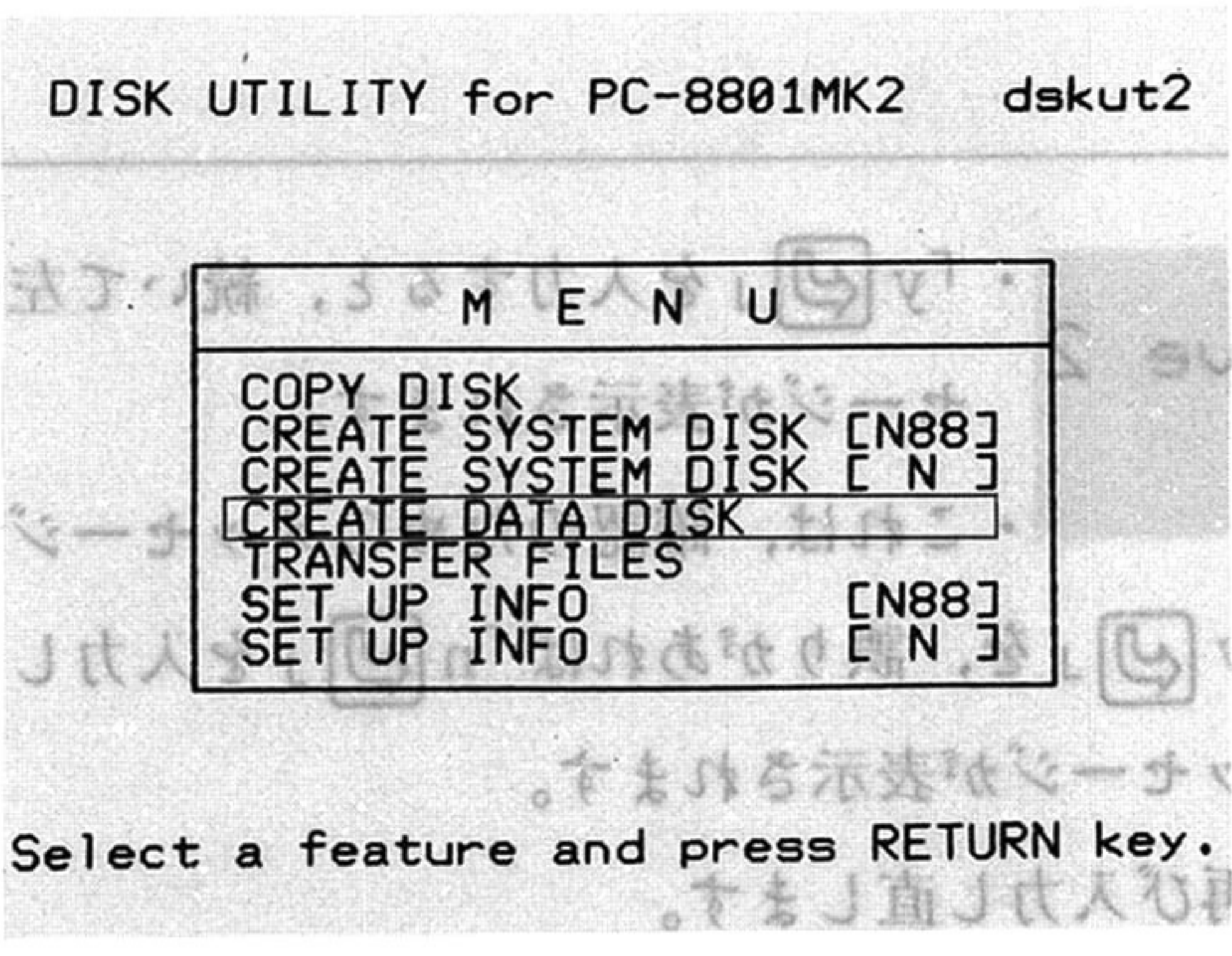
データディスクを作るには，システムディスクに入っている「dskut2.n88」というプログラムを使います。

「dskut2.n88」の使い方は次のとおりです。

①　ユーティリティプログラム(「dskut2.n88」)の読み込み

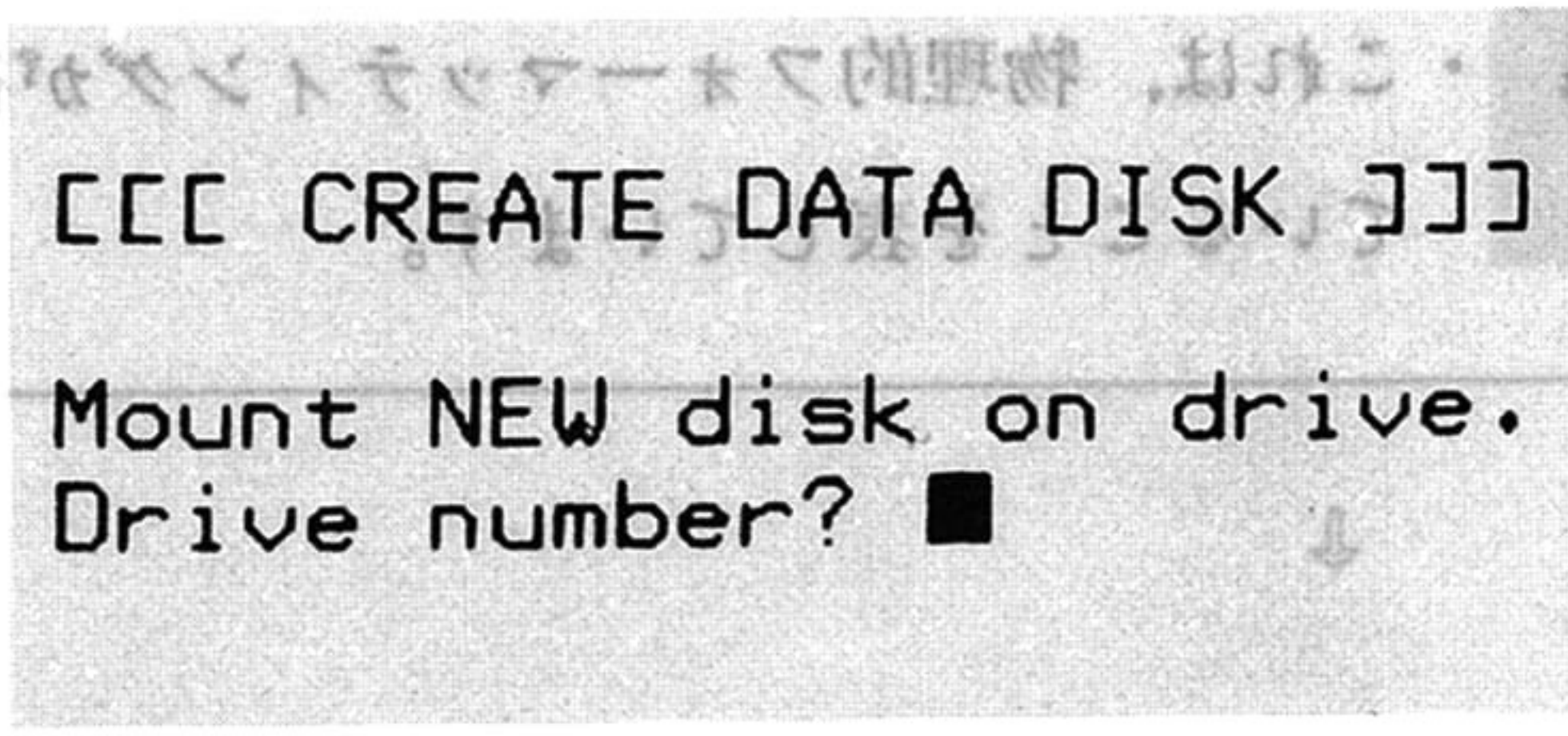
```
load "dskut2.n88"
Ok
```

・次のコマンドをキーインします。

load　"dskut2.n88" ⏎

・するとフォーマッティングのプログラムがメモリーに読み込まれます。

⇩

② ユーティリティプログラムの実行

```
run
```

・誤操作などでシステムディスクを壊すことがないように，システムディスクが入っているドライブのレバーを上げておきます。

・ドライブ2に市販のディスクをセットし，[f・5]キーを押します。

・[f・5]キーには「run$^{C}_{R}$」というコマンドが入っているので，「run [↵]」とキーインしても同じ結果になります。

⇩

③ 処理の選択

```
DISK UTILITY for PC-8801MK2    dskut2

          M  E  N  U
    COPY DISK
    CREATE SYSTEM DISK  [N88]
    CREATE SYSTEM DISK  [ N ]
    CREATE DATA DISK
    TRANSFER FILES
    SET UP INFO         [N88]
    SET UP INFO         [ N ]

Select a feature and press RETURN key.
```

・[f・5]キーを押すと画面が消去(クリア)され，ユーティリティプログラムの実行が始まります。

・画面は左のようなメニューになります。

・このとき，[↑][↓]キーを使って「CREATE DATA DISK」を選択し，[↵]キーを押します。

(注) [　] で囲まれた文字列は，そこが反転した状態であることを示します。

⇩

④ ドライブの指定

```
[[[ CREATE DATA DISK ]]]

Mount NEW disk on drive.
Drive number? ■
```

・画面がクリアされ，左のメッセージが表示されます。

・このとき，市販のディスクをセットしたドライブ番号を入力します。ここでは「2 [↵]」とキーインします。

⇩

⑤　物理的フォーマッティングの指定

```
Do you need physical formatting(y/n)? ■
```

・ドライブ番号を入力すると，続いて上のメッセージが表示されます。

・これは，物理的フォーマッティングを行うか否かを聞いています。

・フォーマッティングを行うフロッピーディスクが購入したばかりのものならば「y ⏎」を，すでに一度使用したものならば「n ⏎」を入力します。

・「y ⏎」を入力すると，⑥のメッセージが表示されます。

・「n ⏎」を入力すると，⑧のメッセージが表示されます。

・物理的フォーマッティングとは，フロッピーディスクにトラック番号やセクタ番号を書き込む作業のことです。

⇩　「y ⏎」を入力したとき　　（「n ⏎」を入力したとき　⇨　⑧へ）

⑥　確認のための入力

```
Format a disk on drive 2
Sure(y/n)? ■
```

・「y ⏎」を入力すると，続いて左のメッセージが表示されます。

・これは，確認のためのメッセージです。

・⑤までの入力に誤りがなければ「y ⏎」を，誤りがあれば「n ⏎」を入力します。

・「y ⏎」を入力すると，⑦のメッセージが表示されます。

・「n ⏎」を入力すると，⑤から再び入力し直します。

⇩　「y ⏎」を入力したとき　　（「n ⏎」を入力したとき　⇨　⑤へ）

⑦　フォーマッティングの実行(1)

```
Physical formatting.
```

・続いて，左のメッセージが表示されます。

・これは，物理的フォーマッティングが行われていることを表しています。

⇩

⑧　フォーマッティングの実行(2)

```
System formatting.
```

・続いて，左のメッセージが表示されます。これは，システムフォーマッティングが行われていることを表しています。

・システムフォーマッティングとは，ファイル管理領域の初期化を行う作業のことです。

⑨ 終了のメッセージ

```
Completed.
Ok
```

・左のメッセージが表示されたら，ディスクのフォーマッティングは終了します。

これで，市販のディスクがデータディスクとして使えるようになります。フォーマッティングの実行結果例は，次のとおりです。

・物理的フォーマッティングを行う場合

```
[[[ CREATE DATA DISK ]]]

Mount NEW disk on drive.
Drive number? 2

Do you need physical formatting(y/n)? y
Format a disk on drive 2
Sure(y/n)? y

Physical formatting.

System formatting.

Completed.
Ok
```

・物理的フォーマッティングを行わない場合

```
[[[ CREATE DATA DISK ]]]

Mount NEW disk on drive.
Drive number? 2

Do you need physical formatting(y/n)? n

System formatting.

Completed.
Ok
```

1.4.2 システムディスクとデータディスクの違い

フロッピーディスクには，N_{88}DISK-BASICを起動させるためのシステムディスクと，プログラムやデータファイルを記録するためのデータディスクがあります。 両方とも同じミニ・フロッピーディスクなので外見上の違いはありませんが，中に記録されている情報に違いがあります。

システムディスクにはN_{88}DISK-BASICを動かすための「システムプログラム」が記録されていて，データディスクにはそれがありません。

システムディスクとデータディスクの使用区分を下の図に示しておきます。

システムディスク	システム領域	ユーザ領域	ファイル管理領域	ユーザ領域

データディスク	ユーザ領域	ファイル管理領域	ユーザ領域

〈一口メモ〉

ユーティリティプログラム

システムソフトウェアの一部としてあらかじめ用意されていて，利用者のコンピュータ作業の補助用に作られている独立したプログラムのことを「ユーティリティプログラム」といいます。

PC-8801mkIIのユーティリティプログラムには，ディスクの内容を別のディスクへコピーするプログラムや，メモリーの内容をプリンタに出力するプログラムなどがあります。

1.4.3 ディスクのコピー方法

(1) あらまし

システムディスクには，システムディスクやデータディスクの内容をそっくりそのまま別のディスクにコピーするユーティリティプログラムも用意されています。

ディスクの内容を別のディスクにそのままコピーすることを，特に「バックアップ」と呼ぶこともあります。

(2) システムディスクのコピー

操作中に誤ってシステムディスクを壊してしまうと困るので，別のディスクに内容をコピーしてそのディスク(バックアップ・シート)を使うようにしましょう。

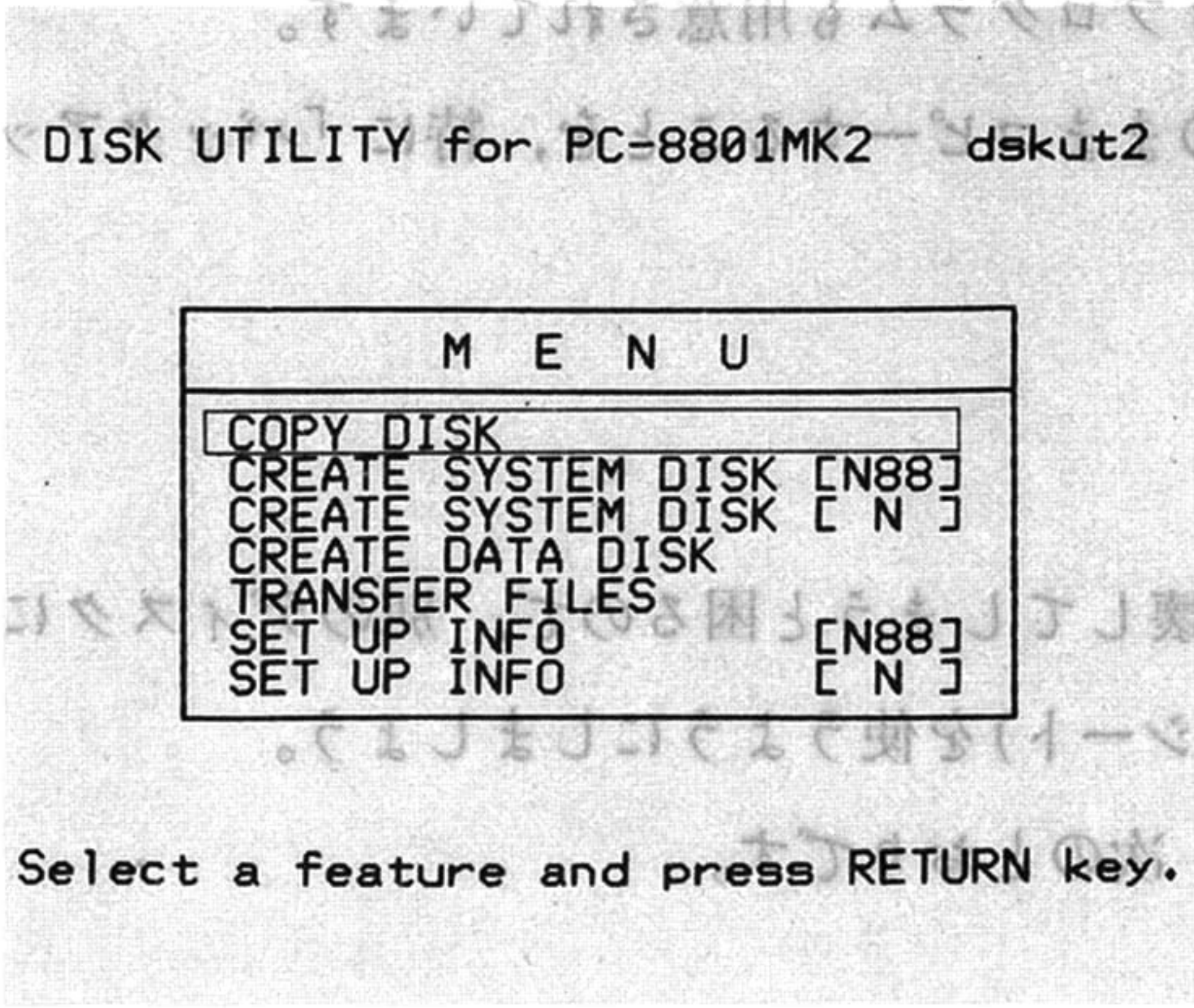
システムディスクのコピーの方法は，次のとおりです。

① ユーティリティプログラム(「dskut2.n88」)の読み込み

```
load "dskut2.n88"
Ok
```

・ドライブ1にシステムディスクをセットした後，「dskut2.n88」をメモリーに読み込みます。

⇩

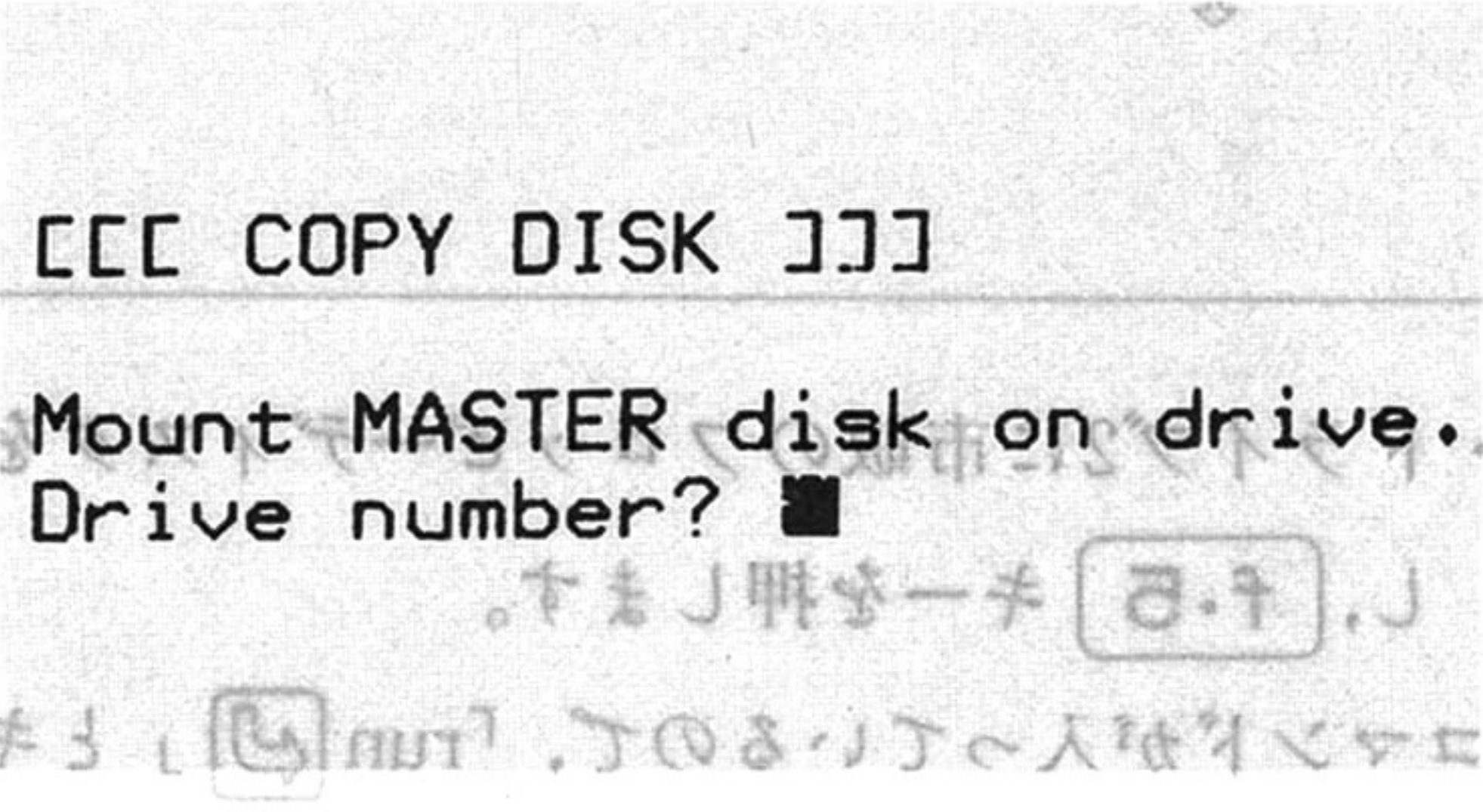
② ユーティリティプログラムの実行

```
run
```

・ドライブ2に市販のフロッピーディスクをセットし，[f･5]キーを押します。

・[f･5]キーには「run$^{C}_{R}$」というコマンドが入っているので，「run[↵]」とキーインしても同じ結果になります。

⇩

③ 処理の選択

```
DISK UTILITY for PC-8801MK2    dskut2

        M  E  N  U
   COPY DISK
   CREATE SYSTEM DISK [N88]
   CREATE SYSTEM DISK [ N ]
   CREATE DATA DISK
   TRANSFER FILES
   SET UP INFO        [N88]
   SET UP INFO        [ N ]

Select a feature and press RETURN key.
```

・[f･5]キーを押すと，画面が消去(クリア)され，ユーティリティプログラムの実行が始まります。

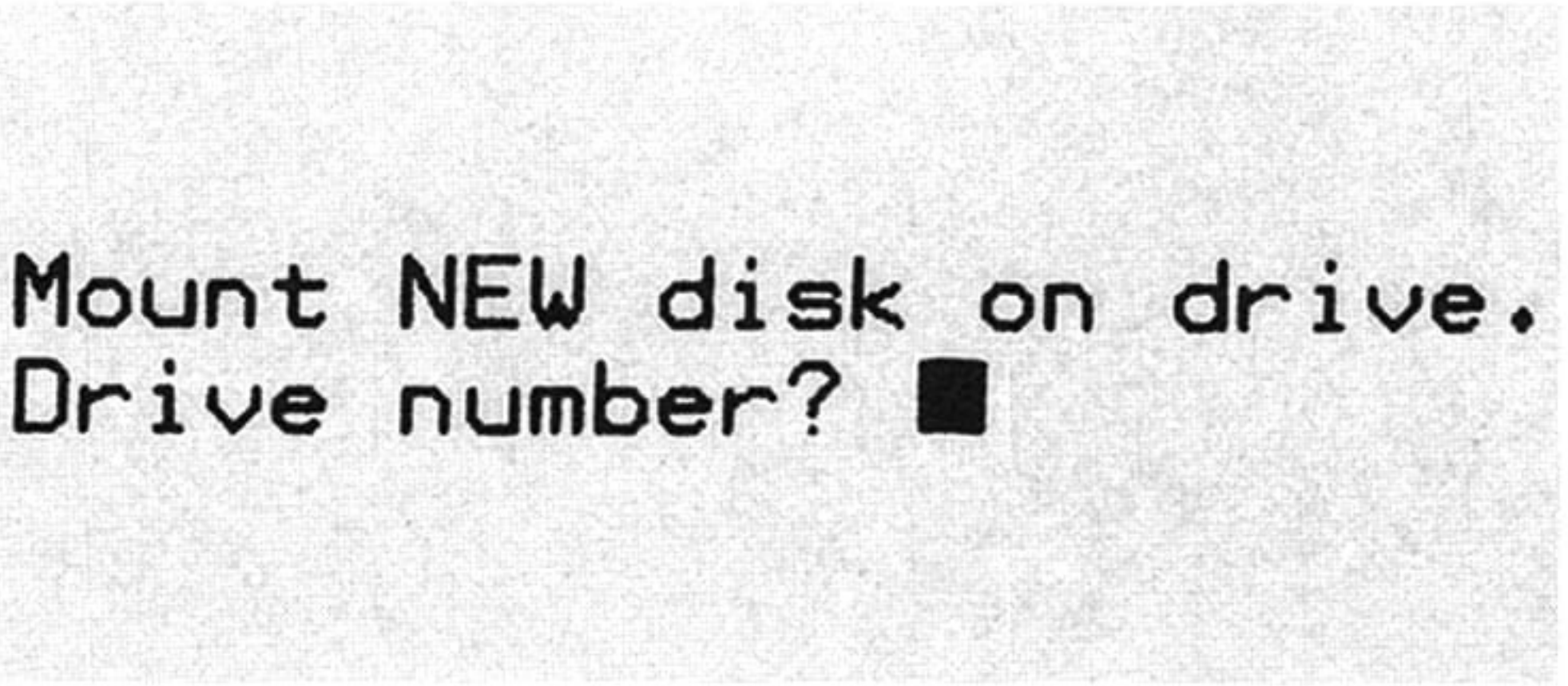
・画面は左のようなメニューになります。

・このときすでに「COPY DISK」が選択されているので，そのまま[⏎]キーを押します。

(注) [ ]で囲まれた文字列は，そこが反転した状態であることを示します。

⇩

④ ドライブの指定(1)

```
[[[ COPY DISK ]]]

Mount MASTER disk on drive.
Drive number? ■
```

・画面がクリアされ，左のメッセージが，表示されます。

・これは，コピーするディスクがセットされているドライブの番号を指定しなさいというメッセージです。

・ここでは，システムディスクがセットされているドライブ1を指定して，「1[⏎]」とキーインします。

⇩

⑤ ドライブの指定(2)

```
Mount NEW disk on drive.
Drive number? ■
```

・「1[⏎]」と入力すると，続いて左のメッセージが表示されます。

・これは，コピーする先のディスクがセットされているドライブの番号を指定しなさいというメッセージです。

・ここでは，市販のディスクをドライブ2にセットして「2[⏎]」とキーインします。

⇩

⑥ 物理的フォーマッティングの指定

```
Do you need physical formatting(y/n)? ■
```

・「2⏎」と入力すると，続いて上のメッセージが表示されます。

・これは，物理的フォーマッティングを行うか否かを聞いています。

・フォーマッティングを行うフロッピーディスクが購入したばかりのものならば「y⏎」を，すでに一度使用したものならば「n⏎」を入力します。

・「y⏎」を入力すると⑦のメッセージが表示され，「n⏎」を入力すると⑨のメッセージが表示されます。

⇩　「y⏎」を入力したとき　　（「n⏎」を入力したとき ⇨ ⑨へ）

⑦ 確認のための入力

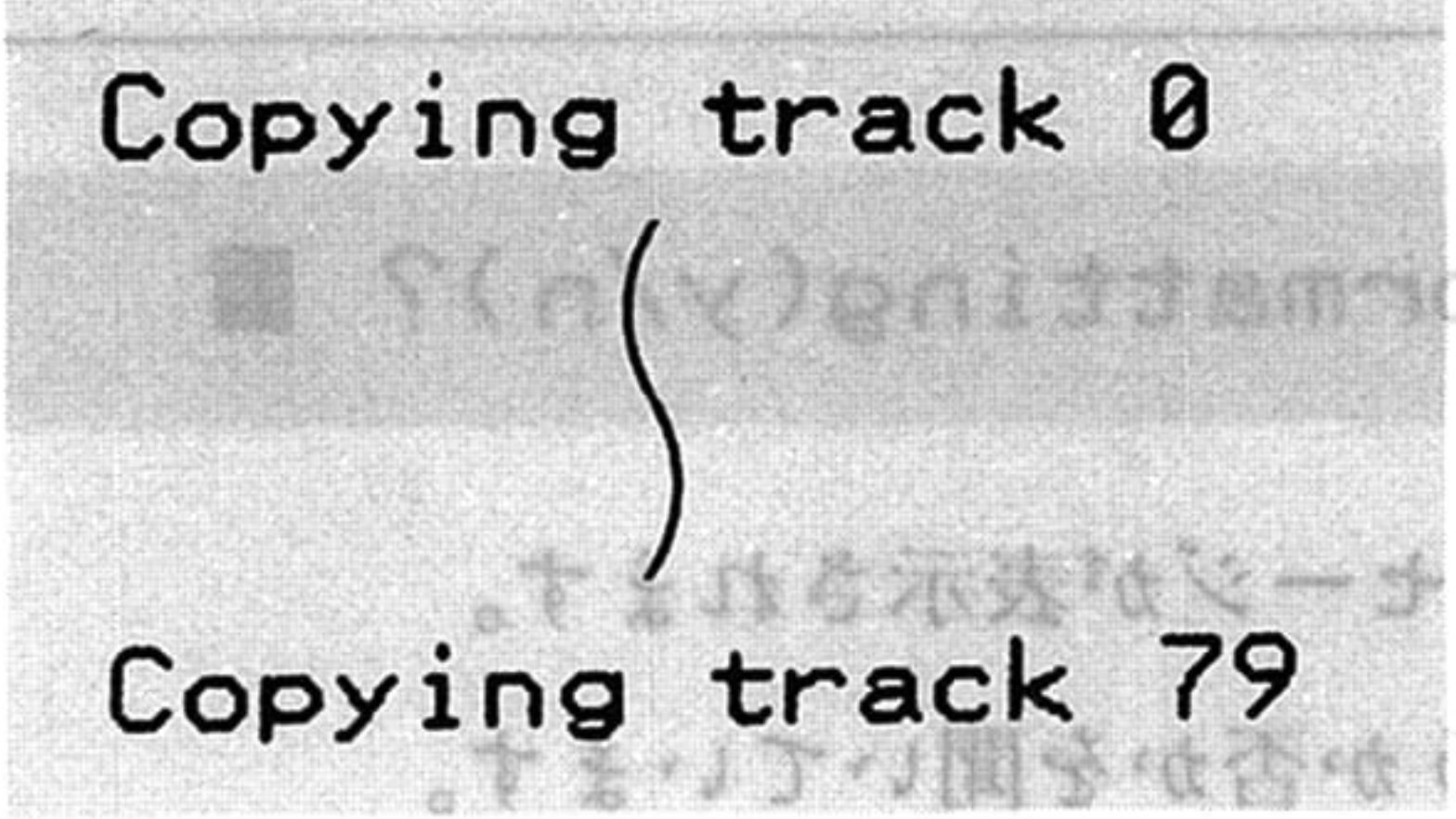
```
Format a disk on drive 2
Sure(y/n)? ■
```

・「y⏎」を入力すると，続いて左のメッセージが表示されます。

・これは，確認のためのメッセージです。

・⑥での入力に誤りがなければ「y⏎」を，誤りがあれば「n⏎」を入力します。

・「y⏎」を入力すると⑧のメッセージが表示され，「n⏎」を入力すると⑥から再び入力し直します。

⇩　「y⏎」を入力したとき　　（「n⏎」を入力したとき ⇨ ⑥へ）

⑧ フォーマッティングの実行

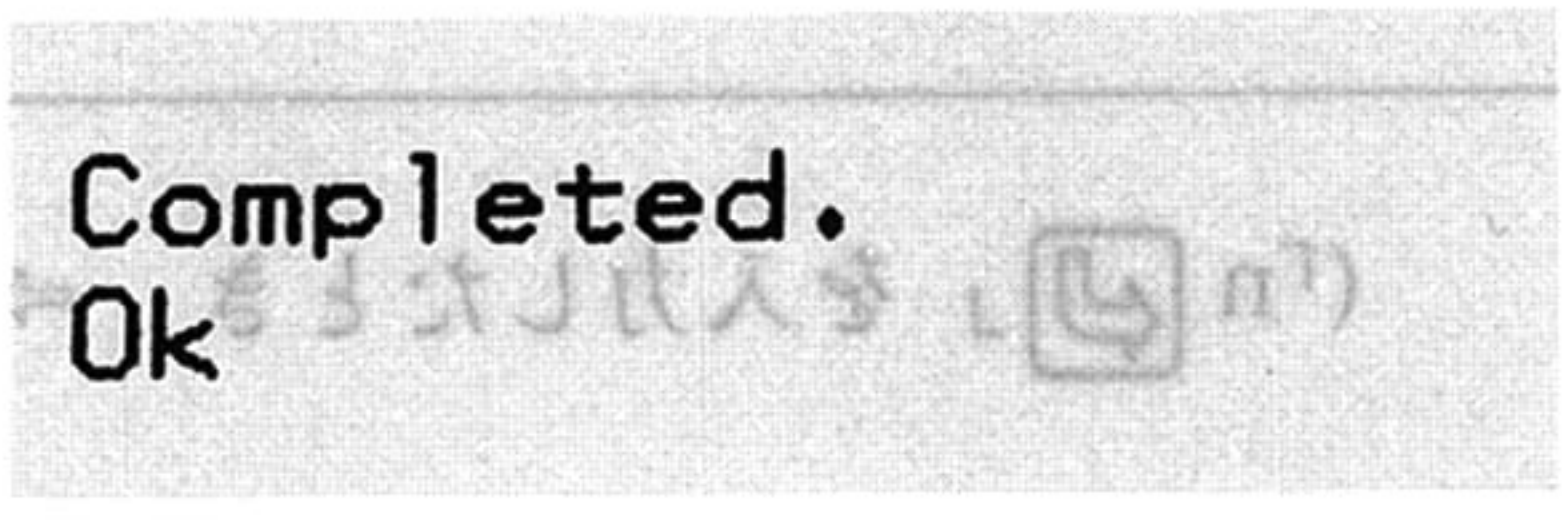
```
Physical formatting.
```

・続いて，左のメッセージが表示されます。

・これは，物理的フォーマッティングが行われていることを表しています。

⇩

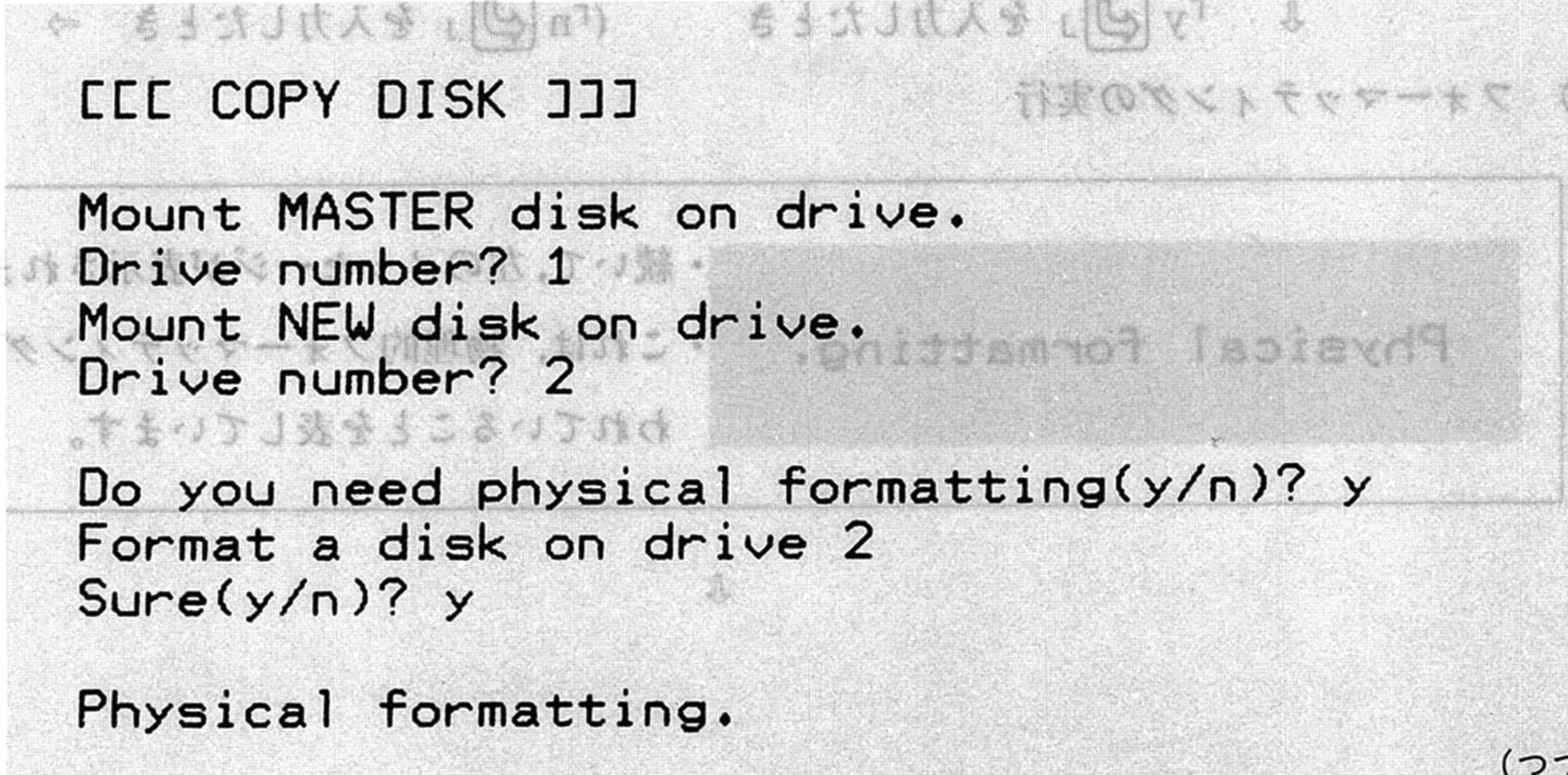
⑨　コピーの実行

```
Copying track 0
  (
   )
Copying track 79
```

・続いて，左のメッセージが順次表示されます。

・これは，現在どのトラックをコピーしているかを表しています。

⇩

⑩　終了のメッセージ

```
Completed.
Ok
```

・左のメッセージが表示されたら，ディスクのコピーは終了します。

これで，システムディスクのコピーが終了しました。これからはコピーしたシステムディスクを使うようにして，オリジナルのシステムディスクはしまっておくようにしましょう。

なお，コピーしたシステムディスクのライトプロテクトノッチに銀紙(ライトプロテクトシール)をはっておくと，より安全です。

コピーの実行結果例は次のとおりです。

・物理的フォーマッティングを行う場合

```
[[[ COPY DISK ]]]

Mount MASTER disk on drive.
Drive number? 1
Mount NEW disk on drive.
Drive number? 2

Do you need physical formatting(y/n)? y
Format a disk on drive 2
Sure(y/n)? y

Physical formatting.
```

(つづく)

```
Copying track 0
        (
        )
Copying track 79

Completed.
Ok
```

・物理的フォーマッティングを行わない場合

```
[[[ COPY DISK ]]]

Mount MASTER disk on drive.
Drive number? 1
Mount NEW disk on drive.
Drive number? 2

Do you need physical formatting(y/n)? n

Copying track 0
        (
        )
Copying track 79

Completed.
Ok
```

(注) PC-8801mkII Model20では,「dskutl.n88」というユーティリティプログラムを使ってコピーします。

2章　プログラムの作成手順

PC-8801mkIIに一連のまとまった仕事をさせるには，プログラムを入力して実行します。プログラムには，市販されているもの，ソフトウェア・ハウスなどに注文したもの，自分で作ったものなどいろいろあります。

ここでは「プログラムとはどんなものか」，「プログラムを作るにはどうするか」，「プログラムを実行するにはどうするか」などを基本に学習します。

2.1　基本的な命令のいろいろ

まず，プログラムを作成していく前に知っていた方がよい基本的な命令を，いくつか説明します。

2.1.1　メモリーのプログラムを消す

現在メモリーには「dskut2.n88」というプログラムが記憶されています。そのプログラムを消すには「NEW」コマンドを使います。NEWコマンドの一般形式と書き方例を次に示します。

(NEWコマンドの説明)

＜一般形式＞

NEW

・メモリーに記憶しているプログラムやデータを消し去ります。

＜書き方例＞

・　NEW

設問 2　今メモリーに記憶しているプログラムを消して下さい。

【 実行結果例 】

```
new
Ok
```

【 実行後の説明 】

・キーボードから「NEW」とキーインした後 ↵ キーを押します。

・これでメモリーに記憶されているプログラムはすべて消えてしまいました。

・プログラムが消えているかどうかの確認は，「2.1.3　プログラムを作るために」で説明します。

2.1.2　結果を表示する命令

（1）　演算結果の表示

計算した結果を画面に表示するには，「PRINT」命令を使います。演算結果を画面に表示する命令の一般形式と書き方例は次のとおりです。

(PRINT命令の説明)

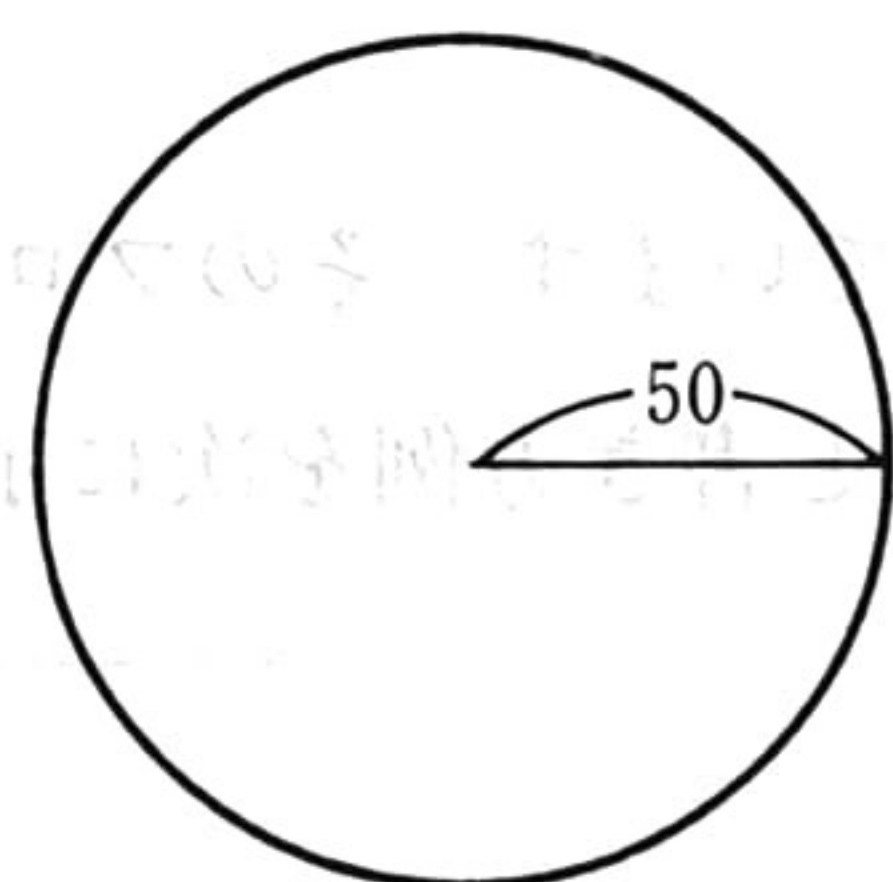
＜一般形式＞

PRINT　数式

・画面に数式の演算結果を表示します。

＜書き方例＞

・　PRINT　1+2+3　　　数式「1+2+3」の演算結果を画面に表示します。

設問 3　半径50の円の円周を求めて下さい。

（ただし，$\pi=3.14$ とします。）

【 実行結果例 】

```
print 2*3.14*50
 314
Ok
```

【 実行後の説明 】

・キーボードから「PRINT　2*3.14*50」とキーインした後 ↵ キーを押します。

・「*」という記号は，乗算(掛け算)のときに「×」の代わりに使います。

・「314」の前の一文字の空白は符号を表示するためのもので，演算結果が負の値のときはここに「－」が表示されます。

・また，「PRINT」命令の代わりに「？」を使っても，結果は同じになります。

(2) 四則演算

PC-8801mkIIは，通常の数式と同じような書き方で演算の式を書くことができます。ただし，演算で使う記号(「＋」，「－」等)は一部異なることがあります。

通常の数式とPC-8801mkIIの演算式との対応は，次のとおりです。

	通常の数式	PC-8801mkIIの数式
加算	＋ (例) 1＋1	＋ (例) 1＋1
減算	－ (例) 2－1	－ (例) 2－1
乗算	× (例) 5×3	＊ (例) 5＊3
除算	÷ (例) 10÷2	／ (例) 10／2
累乗 (べき乗)	X^a (例) 5^2	X^A (例) 5^2
( ) (カッコ)	( ) (例) {(5＋1)×(5－1)}÷2	( ) (例) ((5＋1)＊(5－1))／2

なお，演算の優先順位は次のとおりです。

① ( )　カッコ内の計算
② ^　累乗
③ －　負記号
④ ＊，／　乗算，除算
⑤ ＋，－　加算，減算

同じ優先順位の記号では，先に出てきた方(左側に書いてある方)が優先されます。

(例) 5＊4／2　「5＊4」が先に計算され，次に「20／2」が計算される。

〈一口メモ〉

コマンドレベル

コマンドとは，プログラムの中で使われる命令ではなく，キーボード等から直接与える命令のことを言います。パソコンのコマンドレベルとは，「OK」のメッセージが表示された後，コマンドが与えられるのを待っている状態のことです。

2.1.3　プログラムを作るために

設問2，設問3で学習した「NEW」コマンドや「PRINT」命令は実行した後，メモリーには記憶されません。したがって，同じ命令を繰り返す場合でも、その都度命令を入力し直さなければなりません。

一般に，一つの目的の仕事(処理)をさせる場合いくつかの命令が必要になります。そして，それぞれの命令を処理の手順に沿って実行させます。このように，いくつかの命令を処理の手順に沿って並べたものをプログラムといいます。ところが，このプログラム(一つ一つの命令)をその都度入力して実行するわけにはいきません。

そこで，一つ一つの命令をメモリーに記憶させる方法について学習することにします。

（1）　プログラムと行番号

まずプログラムの例を示します。

```
10 REM ﾌｸﾘ ｹｲｻﾝ ﾌﾟﾛｸﾞﾗﾑ
20 PRINT "--- ｹｲｻﾝ ｶｲｼ ---":PRINT
30 PRINT "ｺﾞｳｹｲ=";100000!*(1+7/100)^5
40 PRINT
50 PRINT "--- ｹｲｻﾝ ｵﾜﾘ ---"
60 END
```

このように，プログラムには命令の先頭に一行ごとに番号が付いています。この番号のことを「行番号」と呼んでいます。

行番号を付けて命令をキーインすると設問3のときとは違って，命令の実行は行われずにメモリーに記憶されます。また，このようにメモリーへ記憶した命令は「NEW」コマンドを実行したり，電源スイッチを切ったりしない限り消えることはありません。

したがって，このようにして作られた命令の集まり(プログラム)は実行コマンド(「2.2.3 作成したプログラムの実行」で説明します)を使って何回でも実行することができるようになります。

なお，行番号は1～65529の範囲で使用でき，プログラムは行番号の小さい順に記憶されます。そして，行番号の小さい順にプログラムを実行して行きます。

また，普通行番号は 10, 20, 30………というように間をあけて付けます。こうしておくと，後から命令を付け加えるために他の行番号を変更しなくても 11, 12, 13………のように追加したい場所に簡単に命令を挿入することができます。

（2）　プログラムを表示する

プログラムを画面に表示するには，「LIST」コマンドを使います。

LISTコマンドの一般形式を次に示します。なお，書き方例は「2.2.2　キーインしたプログラムを画面で見る」を参照して下さい。

(LISTコマンドの説明)

<一般形式>

LIST 開始行番号－終了行番号

- メモリーに記憶しているプログラムを画面に表示します。
- 行番号を一つだけ指定すると，その行だけを表示します。
- 開始行番号を省略すると，プログラムの先頭の行から終了行番号として指定した行までを表示します。
- 終了行番号を省略すると，開始行番号として指定した行からプログラムの最後の行までを表示します。
- 開始行番号と終了行番号の両方を省略すると，プログラムの先頭の行から最後の行まですべて表示します。

設問 4　LISTコマンドを使ってメモリーに記憶しているプログラムをすべて画面に表示して下さい。

【 実行結果例 】

```
list
Ok
```

【 実行後の説明 】

- LISTコマンドは，開始行番号も終了行番号も省略して実行しています。
- LISTコマンドを実行すると何も表示せずにいきなり「Ok」と表示されるのは，メモリー中にプログラムが記憶されていないためです。(設問2でプログラムを消しています。)
- プログラムの表示については，次の項で学習します。

〈一口メモ〉

直接モードと間接モード

パソコンでの命令の実行には「直接モード」と「間接モード」の二つのモードがあります。

直接モードは，命令をキーインしたら直ぐに実行する状態のことを言います。しかしこのモードでは，キーインした命令がメモリーに記憶されません。

間接モードは，命令をキーインしても直ぐには実行せずに，RUNコマンドを与えることによって命令を実行することを言います。このモードでは，キーインする命令の先頭に必ず「行番号」と呼ばれる実行順序を決める番号を付けなければなりません。なお，このモードでキーインした命令はすべてメモリーに記憶されます。

つまり，命令の先頭に行番号を付けてキーインし，RUNコマンドで命令を実行することを「間接モード」といい，行番号を付けないで命令をキーインし，直ぐに実行させることを「直接モード」といいます。

2.2　プログラム例(1)――― 体重の比較をするプログラム

それでは，実際にプログラムを作って行くことにしましょう。ここでは，自分の身長から標準体重を算出し自分の体重と比較する次のようなプログラムを作ることにします。

【 体重比較プログラム 】

```
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 END
```

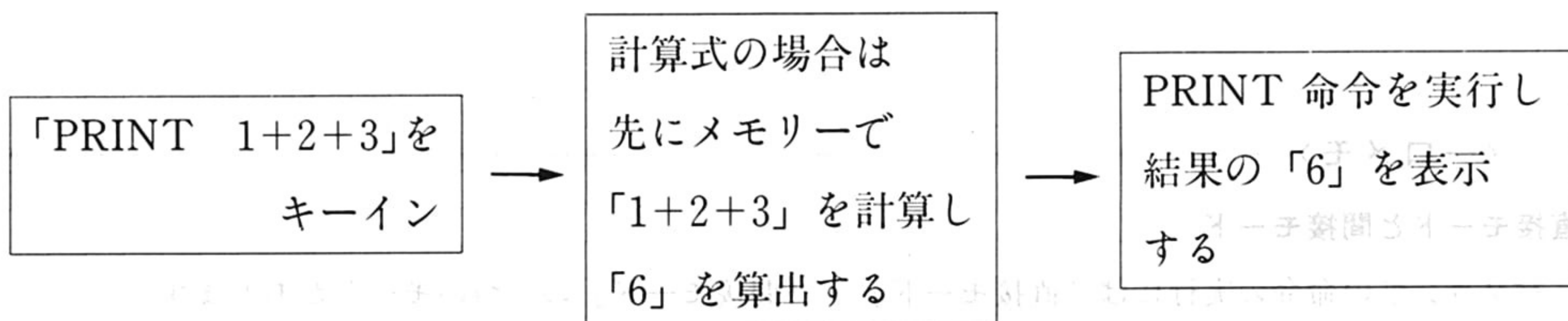
2.2.1　命令の説明

（1）　データの表示

データを画面に表示するには，「PRINT」命令を使います。

前項では演算結果を表示するためにPRINT命令を使いましたが，PRINT命令本来の機能はデータを画面に表示することです。

演算結果を表示する命令を実行すると，PC-8801mkIIは次のような動きをします。

「PRINT　1+2+3」をキーイン → 計算式の場合は先にメモリーで「1+2+3」を計算し「6」を算出する → PRINT 命令を実行し結果の「6」を表示する

PRINT命令の一般形式と書き方例を次に示します。

(PRINT命令の説明)

<一般形式>

PRINT　式または文字式

・画面に文字式や数値を表示します。

・式や文字式を省略すると，1行だけ改行します。

<書き方例>

①	PRINT　3+5	画面に「3+5」の演算結果「8」を表示します。
②	PRINT　HENSU	画面に「HENSU」という変数の内容を表示します。
③	PRINT　"ガメン"	画面に「ガメン」という文字を表示します。
④	PRINT　MOJI$	画面に「MOJI$」という変数の内容を表示します。
⑤	PRINT　,	画面を1行だけ改行します。

(2) 変数

数値や文字などのデータをメモリーに記憶しておくには，データの入れ物が必要になります。BASICでは，この入れ物のことを「変数」と呼んでいます。したがって，変数をたくさん用いることによって，いろいろなデータをメモリーに記憶しておくことができます。

しかし，それぞれの変数に異なる名前を付けておかなければ区別することができません。そこで，それぞれの変数には必ず名前を付けて使います。この名前のことを「変数名」といいます。

また変数には，大きく分けると金額や数量などを表す数値を記憶する「数値型変数」と，商品名や住所・氏名などを表す文字を記憶する「文字型変数」の2種類があり，それらは変数名の付け方で区別します。

文字型変数と数値型変数の区別は，変数名の後に「型宣言文字」と呼ばれる特別な記号を付けることにより行います。

型宣言文字には「$(ダラー)，%(パーセント)，!(エクスクラメーション)，#(ナンバーサイン)」があり，変数名の後に「$」を付けると文字型変数になります。また，「%，!，#」のいずれかを変数名の後に付けると数値型変数になりますが，それぞれの型宣言文字によって次のように区別されています。

% ………… 整数型

! ………… 単精度実数型

………… 倍精度実数型

なお，変数名の後に型宣言文字を付けない場合は，通常「単精度実数型」の変数として扱われます。また，変数の「初期値(プログラムの実行を始めるときの最初の値)」は，文字型

変数が「ヌル(桁数0で何も記憶されていない状態)」に，数値型変数が「ゼロ(0)」になっています。

（3） データの入力

データをキーボードから入力するときには，「INPUT」命令を使います。

INPUT命令の一般形式と書き方例を次に示します。

(INPUT命令の説明)

<一般形式>

INPUT ”プロンプト文” ；|， 変数1，変数2………，変数n

・キーボードから変数へデータを入力します。

・「”」で囲んだプロンプト文を指定すると，入力の案内文として表示します。

・プロンプト文は省略することができます。

・プロンプト文の後に「；」を付けるとメッセージに続けて「？」を表示しますが，「，」を付けると「？」は表示しません。

・変数をいくつか「，」で区切って書くと，複数のデータを入力することができます。

<書き方例>

① INPUT HENSU	キーボードから変数「HENSU」に数値データを入力します。このとき「？」を表示します。
② INPUT ”データ”；A	キーボードから変数「A」に数値データを入力します。そのとき「データ」というメッセージと「？」を表示します。
③ INPUT ”データ”，B＄	キーボードから変数「B＄」に文字データを入力します。そのとき「データ」というメッセージを表示します。
④ INPUT X，Y，Z	キーボードから変数「X，Y，Z」にそれぞれ数値データを入力します。

（4） 画面をクリアする命令

画面に表示されている文字を消す(クリアする)には，「CLS」命令を使います。CLS命令の一般形式と書き方例を次に示します。

(CLS命令の説明)

<一般形式>

CLS 機能

・画面に描かれている文字や図形をクリアします。

・機能は1～3の範囲で指定し，それぞれ次のような働きをします。

CLS 1 ——— テキスト画面(文字を表示する画面)だけをクリアします。

CLS 2 ——— グラフィック画面(図形や漢字を表示する画面)だけをクリアします。

CLS 3 ——— テキスト画面とグラフィック画面の両方をクリアします。

・機能を省略すると「1」を指定したときと同じ働きをします。

<書き方例>

① CLS 1　テキスト画面に書かれている文字をクリアします。

② CLS 2　グラフィック画面に描かれている図形をクリアします。

③ CLS 3　テキスト画面の文字とグラフィック画面の図形の両方をクリアします。

④ CLS　テキスト画面に書かれている文字をクリアします。

(5) 無条件分岐命令

プログラムの実行は行番号の小さい順に行われますが，実際には不都合が起こることがあります。そこで，プログラムの実行の順序を変えてやる必要が出てきます。このことを「分岐」と呼んでいます。

分岐には，「条件分岐」と「無条件分岐」とがあり，条件分岐は「IF…THEN…ELSE」命令を使い，無条件分岐は「GOTO」命令を使います。それでは，先にGOTO命令の一般形式と書き方例を示しておきます。

(GOTO命令の説明)

<一般形式>

GOTO 行番号またはラベル名

・行番号またはラベル名で指定した行へ処理を分岐します。

<書き方例>

① GOTO 30　行番号30の行へ分岐します。

② GOTO *LAVEL　ラベル名「*LAVEL」の行へ分岐します。

(6) 条件分岐命令

プログラムを作っていくと，条件によって処理が異なることがあります。そのようなときには，条件を判断しその結果で処理を変える命令が必要になります。

条件を判断するには「IF…THEN…ELSE」命令(IF文)を使います。IF文の一般形式と書き方例を次に示します。

(IF…THEN…ELSE命令の説明)

＜一般形式＞

IF　条件式　THEN　命令文1　ELSE　命令文2

・条件式の条件を判断して実行する処理を決定します。

・条件が満たされれば命令文1を実行し，満たされなければ命令文2を実行します。

・ELSEとそれに続く命令文2は必要がなければ省略することができます。

・命令文1，命令文2には実行したい命令かまたは行番号(ラベル名)を書きます。

・命令文1，命令文2には，複数の命令を「：」で区切って書くことができます。

・THENに続けてGOTO命令を書くときは，「GOTO」を省略して「THEN　行番号(ラベル名)」と書くことができます。ELSEについても同様です。

＜条件式＞

条件式で条件を判断するときに「関係演算子」が使われます。関係演算子と条件式の例は次の表のとおりです。

(関係演算子一覧)

関係演算子	意　味	条件式の例	条件式の意味
＝	等しい	A－B＝C	A－Bの値がCの値と等しい。
		A＄＝"イロハ"	A＄の内容が「イロハ」と等しい。
＜	小さい	KD＜0	KDの値が0より小さい。(負の数)
		A＜B	Aの値がBより小さい。
＞	大きい	X＋Y＞1000	X＋Yの値が1000より大きい。
		A＞B	Aの値がBの値より大きい。
＜＞ ＞＜	等しくない	A＜＞B A＞＜B	Aの値はBの値と等しくない。
＜＝ ＝＜	小さいかまたは等しい	KS＜＝99 KS＝＜99	KSの値が「99」に等しいかまたは小さい。
＞＝ ＝＞	大きいかまたは等しい	KM＞＝500 KM＝＞500	KMの値が「500」に等しいかまたは大きい。

＜書き方例＞

① IF　A＜B　THEN　100

変数「A」の値が変数「B」の値より小さければ行番号100へ分岐します。

② IF　A＜B　THEN　PRINT　X＄

変数「A」の値が変数「B」の値より小さければ変数「X＄」の内容を表

示します。

③ IF A<B THEN PRINT J : GOTO 20
変数「A」の値が変数「B」の値より小さければ変数「J」の値を表示して行番号20へ分岐します。

④ IF A=B THEN 100 ELSE 20
変数「A」の値と変数「B」の値が等しければ行番号100へ分岐し，等しくなければ行番号20へ分岐します。

〈一口メモ〉

ラベル名

PC-8801mkIIでは，IF…THEN…ELSE, GOTO, LISTなどで，行番号の代わりにラベル名を指定することができます。ただし，このラベル名を使用する場合，次の各事項に注意しなければなりません。

① ラベル名の先頭には必ず「*」を付けます。
② ラベル名は必ず英文字で始まる名前を付けます。
③ ラベル名に使える文字は「英文字」，「数字」，「.」です。
④ BASICの予約語(IF，PRINTなどの命令や関数など)をラベル名の一部として使う場合はかまいませんが，予約語をそのままラベル名として使うことはできません。
⑤ ラベル名の長さは，行番号などの桁数を含んで最大255文字まで書けます。
⑥ ラベル名は必ず行番号のすぐ後に書かなければなりません。

以上の制限を守らないと「Syntax error (? SN Error)」になります。また，ラベル名を二重に定義すると「duplicate label (? DU Error)」になります。

（7） プログラムに注釈を付ける

プログラムの先頭などにそのプログラムの内容を表す注釈文を入れておくと便利です。注釈文を付けるには「REM」命令を使います。REM命令の一般形式と書き方例を次に示します。

（REM命令の説明）

<一般形式>

REM　注釈文

・プログラムに注釈文を書くことができます。

・「REM」の代わりに「'」を使っても同じ働きをします。

<書き方例>

① REM　プログラム1　　　「プログラム1」という注釈文を付けています。

② '　ケイサン　プログラム　　「ケイサン　プログラム」という注釈文を付けています。

（8） プログラムの終了宣言

プログラムの最後にはそのプログラムの実行を終了する「END」命令を書きます。END命令の一般形式と書き方例を次に示します。

（END命令の説明）

<一般形式>

END

・プログラムの実行を終了します。

<書き方例>

・　END

設問 5　30ページの体重比較プログラムをメモリーにキーインして下さい。

【キーイン例】

```
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT"ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
```

（つづく）

```
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150

140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 END
```

【 キーイン後の説明 】

・一行一行のキーインには行番号と命令文をキーインした後，必ず ↵ キーを押します。

・命令のキーインは小文字でもかまいません。

また，プログラムの解説は次のとおりです。

【 プログラムの解説 】

①

```
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
```

・プログラムの内容を表している注釈分です。この命令は書いても書かなくてもプログラムの実行には影響ありませんが，一般にはプログラムの内容を分かりやすくするために使います。

②

```
20 CLS
```

・画面に表示されている文字を消しています。

③

```
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
```

・自分の身長を変数「T」に，自分の体重を変数「W」にそれぞれ入力しています。

④

```
50 PRINT
```

・画面を見やすくするために，一行空けています。

⑤

```
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
```

・身長に対する標準体重を変数「HYOJUN」に，標準体重より5%軽い重さを変数「HU」に，標準体重より5%重い重さを変数「HO」にそれぞれ求めています。

⑥

```
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
```

・行番号90では，変数「HU」と変数「HO」の内容を表示して，標準体重の範囲を示しています。

・行番号100では，変数「W」の内容を表示して自分の体重を示しています。

・PRINT命令で「;(セミコロン)」を使うと，「;」の前と後のデータをつなげて表示します。

⑦

```
110 PRINT
```

・行番号50と同様に，画面を見やすくするために1行空けています。

⑧

```
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
```

・行番号120では，自分の体重が標準体重の下限(変数「HU」)より小さければ，「ヤセテマス」というメッセージを表示し行番号150へ分岐しています。

・行番号130では，自分の体重が標準体重の上限(変数「HO」)より大きければ，「フトッテマス」のメッセージを表示します。

⑨

```
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
```

・行番号120と行番号130の条件に当てはまらず体重が標準の範囲内のときは，行番号140で「ヒョウジュンデス」というメッセージを表示します。

⑩
```
150 END
```

・プログラムを終了します。

2.2.2 キーインしたプログラムを画面で見る

プログラムを実行する前に，正しくキーインされているかどうか画面に表示して確認することにします。

メモリーのプログラムを画面に表示するには「LIST」コマンドを使います。

設問 6　今メモリーにキーインしたプログラムをすべて画面に表示して下さい。

【 実行結果例 】

```
list
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO

100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150

140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 END
Ok
```

〈一口メモ〉

⏎キー（キャリッジリターンキー）を押す意味

コマンドやデータの入力，プログラムのキーインなどで⏎キーを押すのは，プログラムやデータの入力の終わりをパソコンに合図するためです。

【 実行後の説明 】

・プログラムをすべて表示するにはLISTコマンドの行番号の指定を省略して実行します。

・標準体重の計算は体重から100を引いて0.9を掛けています。さらに算出した標準体重の±5%を標準体重の範囲としています。

・今メモリーに記憶しているプログラムを用いて，LISTコマンドの使用例を次に示します。

①

```
list
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 END
Ok
```

②

```
list 30
30 INPUT "ｼﾝﾁｮｳ =";T
Ok
```

③

```
list 30-50
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
Ok
```

④

```
list -40
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
Ok
```

⑤

```
list 100-
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 END
Ok
```

2.2.3 作成したプログラムの実行

それでは，いよいよ今までキーインしてきたプログラムを実行することにしましょう。

メモリーに記憶しているプログラムを実行するには「RUN」コマンドを使います。RUNコマンドの一般形式と書き方例は次のとおりです。

(RUNコマンドの説明)

<一般形式>

RUN　行番号またはラベル名

・メモリーに記憶しているプログラムの実行を開始します。
・行番号またはラベル名を指定すると，その行から実行を開始します。
・行番号またはラベル名は，通常省略します。省略するとプログラムの先頭から実行を開始します。

<書き方例>

① RUN　　　　　　プログラムの先頭から実行を開始します。
② RUN 200　　　　行番号200から実行を開始します。
③ RUN ＊START　　ラベル名「＊START」の行から実行を開始します。

設問 7　今メモリーに記憶しているプログラムを実行させて下さい。

【 実行結果例１ 】

```
run
```

【 実行後の説明1 】

・キーボードから「RUN ⏎」とキーインする，または f･5 キーを押します。 f･5 キーには「run CR」という文字列が入っています。「CR」という文字は ⏎ キーを押したときと同じ働きをします。

・すると画面がクリアされて，画面は次のように変わります。

【 実行結果例2 】

```
シンチョウ =? ■
```

【 実行後の説明2 】

・これは，行番号20のCLS命令で画面がクリアされた後に行番号30のINPUT命令が実行されて表示されたメッセージです。

設問 8 プログラムに従って自分の身長と体重を入力し，標準体重とその評価を表示して下さい。

【 実行結果例 】

```
シンチョウ =? 163 ---------------------------------► 行番号30のINPUT命令
タイジ゛ュウ =? 56 --------------------------------► 行番号40の 〃
                    -------------------------------► 行番号50のPRINT命令
ヒョウシ゛ュン タイジ゛ュウ = 53.865 <--> 59.535 ---► 行番号90の 〃
アナタ ノ タイジ゛ュウ = 56 ------------------------► 行番号100の 〃
                    -------------------------------► 行番号110の 〃
ヒョウシ゛ュンテ゛ス --------------------------------► 行番号140の 〃
Ok
```

【 実行後の説明 】

・「シンチョウ ＝？」と表示された後，自分の身長をcm単位で入力します。そのとき，データをキーインした後必ず ⏎ キーを押します。

・続いて「タイジュウ ＝？」と表示された後，自分の体重をkg単位で入力します。

・すると，標準体重,キーインした自分の体重,評価を表示します。

・標準体重と自分の体重の差によって評価のメッセージは変わります。

・このプログラムは，RUNコマンドを使えば，何度でも実行できます。

2.3 プログラムの保存と呼び出し

今まで苦労してキーインしたプログラムもそのままメモリーに記憶させておくと，メモリークリア（プログラムを消す）を行ったり，電源を切ったりしたときに消えてしまって残りません。そこで，フロッピーディスクにプログラムを保存しておくことが必要になります。一度保存しておれば，メモリーをクリアしてしまっても必要に応じてフロッピーディスクから呼び出すことによって何回でも利用することができます。

2.3.1 プログラムに名前を付ける

PC-8801mk IIは，プログラムをフロッピーディスクに保存したり，フロッピーディスクから呼び出したりするための管理をプログラムファイルの名前で行っています。プログラムファイルは「ファイルディスクリプタ」で名前を付けます。

ファイルディスクリプタは次のような形式になっています。

（ファイルディスクリプタの説明）

<一般形式>

”デバイス名 ： ファイル名 ． 拡張ファイル名 ”

・デバイス名は使用する装置の名前を指定しますが，フロッピーディスク装置の場合は特にドライブ番号で指定します。省略すると，「1」を指定したものと見なします。

・ファイル名は6文字以内の任意の文字で指定します。ただし，ファイル名の一部として「：」と「．」は使うことができません。

・拡張ファイル名は一般にファイルの性質を表すために用いますが，ファイル名の一部として使うこともできます。なお，不要なときは省略します。

<書き方例>

① ”2：PROG.001”　　ドライブ2の「PROG.001」というプログラムを意味します。

② ”2：JINJI.P01”　　①と同様の形式ですが，拡張ファイル名でプログラムがデータかを区別しています。

③ ”2：JINJI.D01”　　②と同様です。

④ ”2：KYUYO”　　拡張ファイル名を省略した例です。

⑤ ”2：KYUYOMAST”　　拡張ファイル名をファイル名の一部として利用した例で，初めの6文字がファイル名になり，残りの3文字が拡張ファイル名になります。

⑥ ”KYUYOMAST”　　⑤と同様の意味ですが，ドライブ番号「1」を指定したときと同じになります。

2.3.2 プログラムの保存方法

メモリーに記憶されているプログラムをフロッピーディスクへ保存するには,「SAVE」コマンドを使います。SAVEコマンドの一般形式と書き方例を次に示します。

(SAVEコマンドの説明)

<一般形式>

SAVE　ファイルディスクリプタ

・ファイルディスクリプタで指定したドライブのフロッピーディスクに，指定した名前でプログラムを保存します。

<書き方例>

・ SAVE "2：KYOIKU"　ドライブ2のフロッピーディスクへ「KYOIKU」という名前のプログラムを保存しています。

設問 9　今メモリーに記憶されているプログラムを「TAIJU.1」という名前でドライブ2のフロッピーディスクに保存して下さい。

【 実行結果例 】

```
save "2:TAIJU.1"
Ok
```

【 実行後の説明 】

・「1.4.1　データ用のディスクをつくる方法」で作成したデータディスクをドライブ2にセットした後SAVEコマンドを実行します。

2.3.3 プログラムの呼び出し方法

フロッピーディスクに保存されているプログラムをメモリーへ呼び出す(メモリーに移す)には,「LOAD」コマンドを使います。LOADコマンドの一般形式と書き方例は次のとおりです。

(LOADコマンドの説明)

<一般形式>

LOAD　ファイルディスクリプタ

・ファイルディスクリプタで指定したドライブのフロッピーディスクから，指定した名前のプログラムをメモリーに呼び出します。

<書き方例>

・ LOAD "2：KYOIKU"　ドライブ2のフロッピーディスクから「KYOIKU」という名前のプログラムをメモリーに呼び出しています。

設問 10　今メモリーに記憶されているプログラムを消去して下さい。

【 実行結果例 】

```
new
Ok
list
Ok
```

【 実行後の説明 】

・NEWコマンドを実行してメモリークリアを行っています。

・その後，本当にメモリークリアが行われたかどうかLISTコマンドを使って確かめています。

・LISTコマンドを実行した後すぐに「Ok」が表示されてコマンドレベルに戻っていることで，メモリークリアが行われプログラムが消えていることがわかります。

設問 11　ドライブ2のフロッピーディスクから「TAIJU. 1」という名前のプログラムをメモリーへ呼び出して下さい。

【 実行結果例 】

```
load "2:TAIJU.1"
Ok
```

【 実行後の説明 】

・もし，このときにプログラム名を「TAIJU.1」のように1文字でも間違えたら全く別のプログラムの名前として取り扱われて，次のようなエラーが発生します。そのときは再び正しい名前をキーインし直します。

```
load "2:TAIJU1"
File not found
Ok
```

設問 12　プログラムが正しくロード(LOAD)されているかどうか，プログラムリストをすべて画面に表示して下さい。

【 実行結果例 】

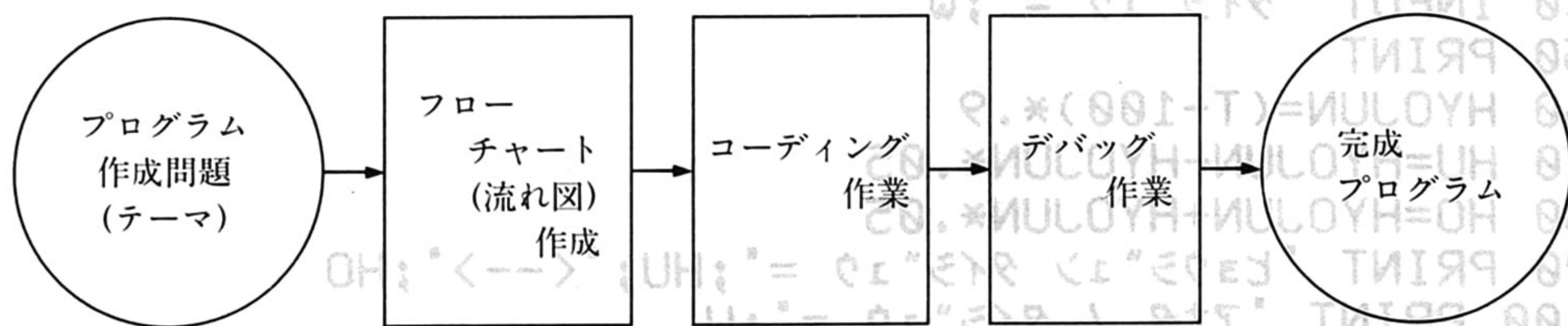
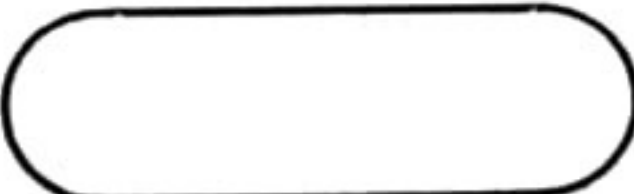
```
list
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 END
Ok
```

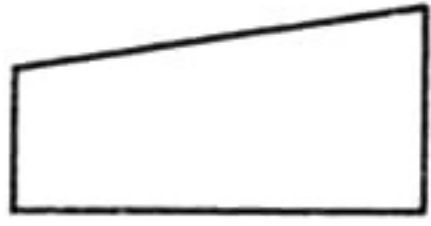
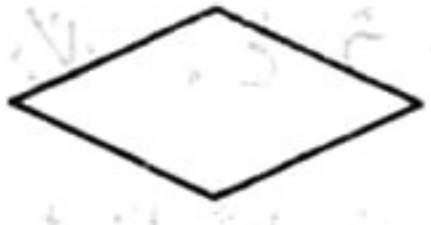
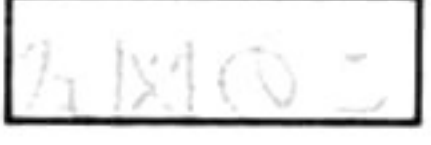
2.4　フローチャート

2.4.1　プログラムの流れを示すために

パソコンに何か処理をさせようとしてすぐにそのプログラムが作れるかというと，必ずしもそうではありません。普通，最初に「どういう処理をどんな順序で実行すれば答えが求まるか」という問題についての処理の流れを図式で表し，プログラム全体を把握します。この図式のことを「フローチャート(Flowchart，流れ図)」と呼びます。フローチャートが完成したら，これに基づいてBASICでプログラムを書いて行きます。

また，フローチャートからコンピュータが理解できる言葉(PC-8801mkIIではN_{88}-BASIC)に書き直すことを「コーディング(Coding)」作業と呼んでいます。フローチャートとコーディングが終わったら，プログラムが正しく作られているかどうかを調べるために実行してみます。処理の手順を間違えたり，パソコンが理解できない言葉で命令文を書いたりしていると，答えが正しく出てきません。このようなプログラムを「バグ(Bug，虫)のあるプログラム」といい，正しい答えが出るようにプログラムの中のバグの原因をよく調べて修正します。このことを「デバッグ(Debug，虫取り)」作業と呼びます。デバッグ作業まで終了すると，プログラムは完成したことになります。

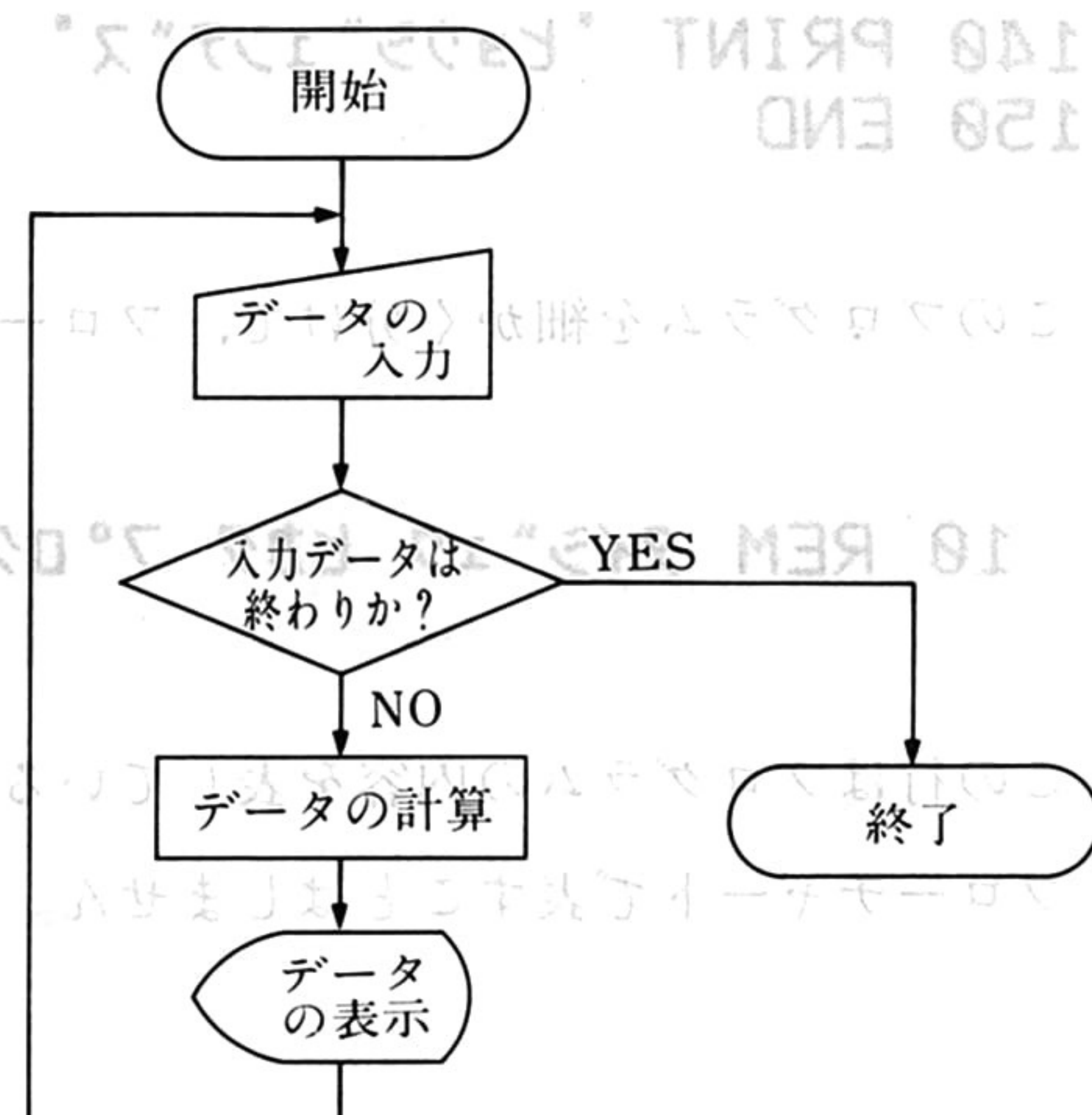
プログラム作成問題(テーマ) → フローチャート(流れ図)作成 → コーディング作業 → デバッグ作業 → 完成プログラム

(フローチャートがしっかりできていないとよいプログラムとは呼べません。)

2.4.2　フローチャートの書き方

(1)　あらまし

右の例はフローチャートの例ですが，フローチャートとは，この例のようにあらかじめ決められている記号を使ってプログラム全体の流れを表すものです。

開始 → データの入力 → 入力データは終わりか？ (YES → 終了) NO → データの計算 → データの表示 → (データの入力へ戻る)

それぞれの記号の使い方は，次のとおりです。

①　（端子記号）　プログラムの開始と終了を表します。プログラムは開始したら必ず終了になるようにします。

② ↓ 処理の流れ(順序)を示します。

③ キーボードからのデータのインプット(Input，入力)を表します。

④ 処理の分岐を表します。条件によって処理を選択するときに使います。

⑤ 処理の機能を表します。メモリー内での演算などに使います。

⑥ 画面表示を表します。データを画面に表示するときに使います。

(2) 作成したプログラムのフローチャート

それでは，いままでキーインしてきたプログラムのフローチャートを表してみましょう。

前項までにキーインしたプログラムリストを次に示します。

【キーインしたプログラムのリスト】

```
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 END
```

このプログラムを細かく分けて，フローチャート記号で表してみることにします。

①

```
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
```

・この行はプログラムの内容を表している注釈文です。注釈文は処理を行っていないので，フローチャートで表すことはしません。

②

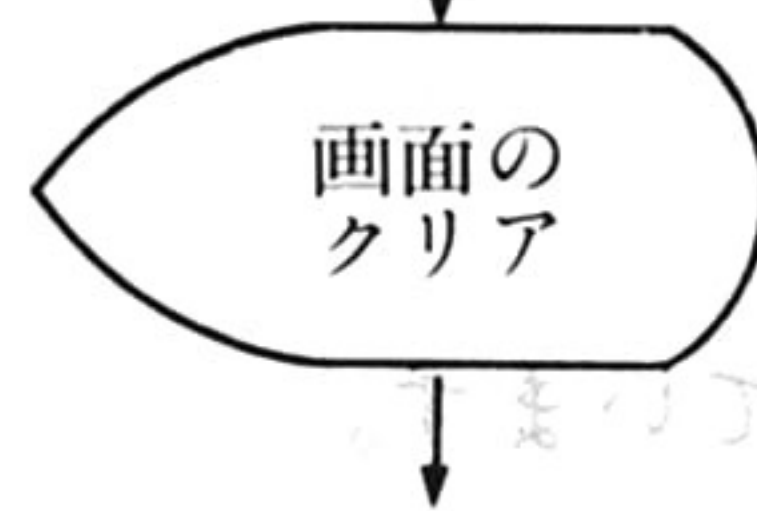
```
20 CLS
```

・この行では画面のクリアを行っています。

・画面のクリアは次のように表します。

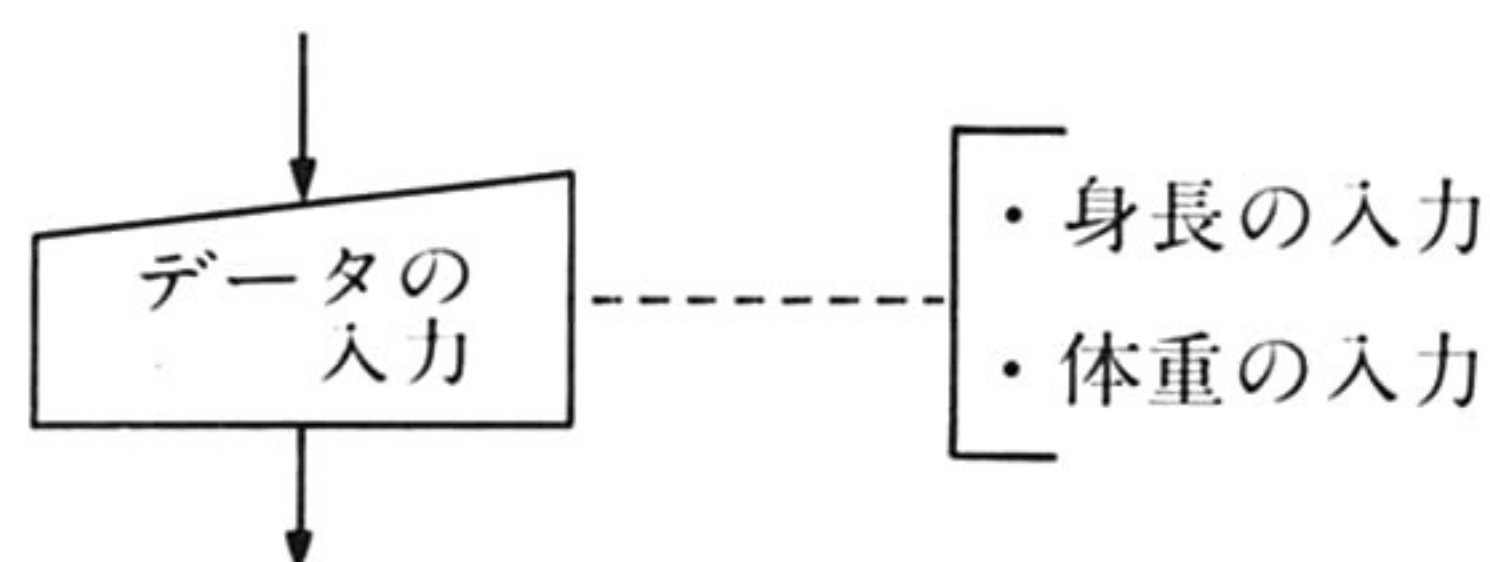
画面の
クリア

③

```
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
```

・この二つの行は，データの入力を行っています。

・データの入力は次のように表します。

データの
入力

・身長の入力
・体重の入力

・［ は，処理に対する注意などを表す「コメント」の記号です。

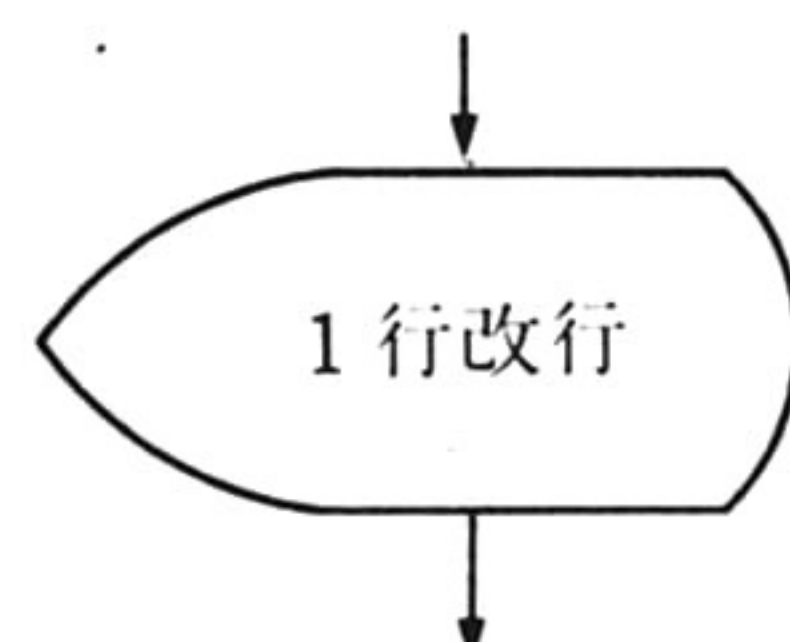
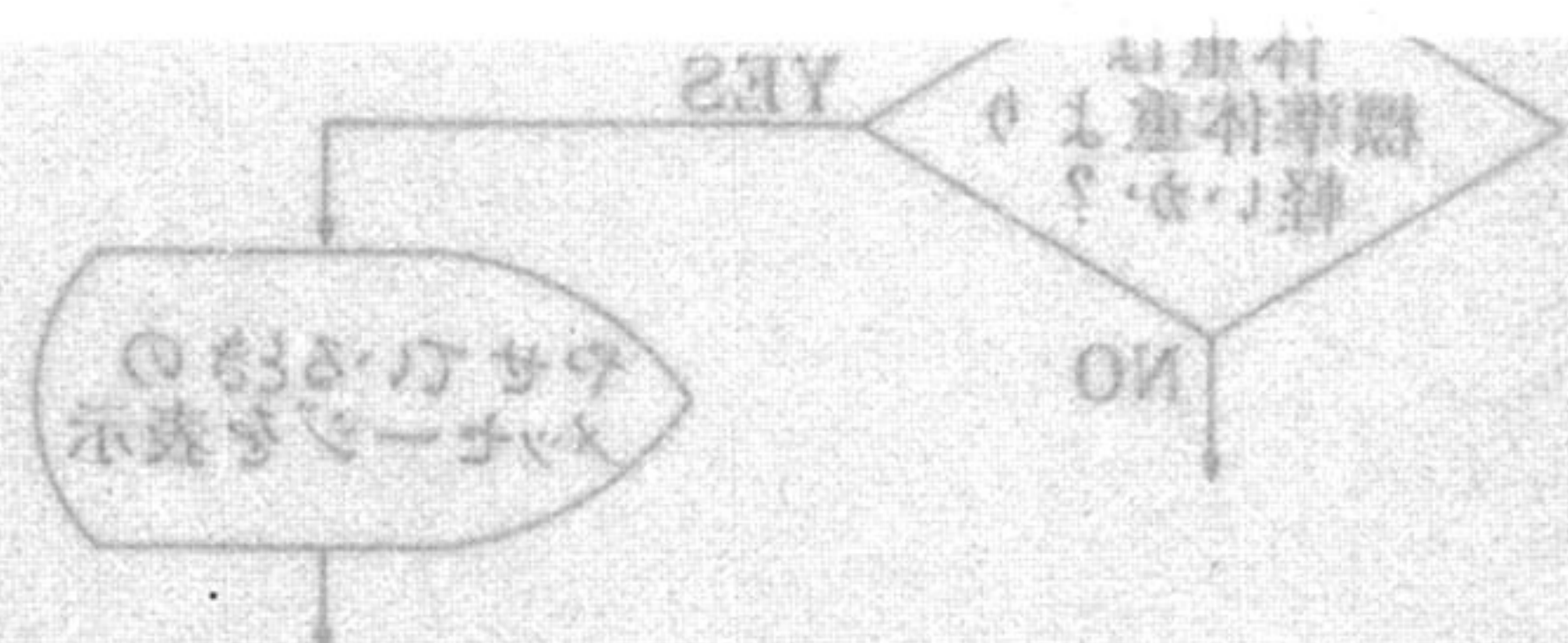
④

```
50 PRINT
```

・この行は，1行の改行を行っています。

・この処理は次のように表します。

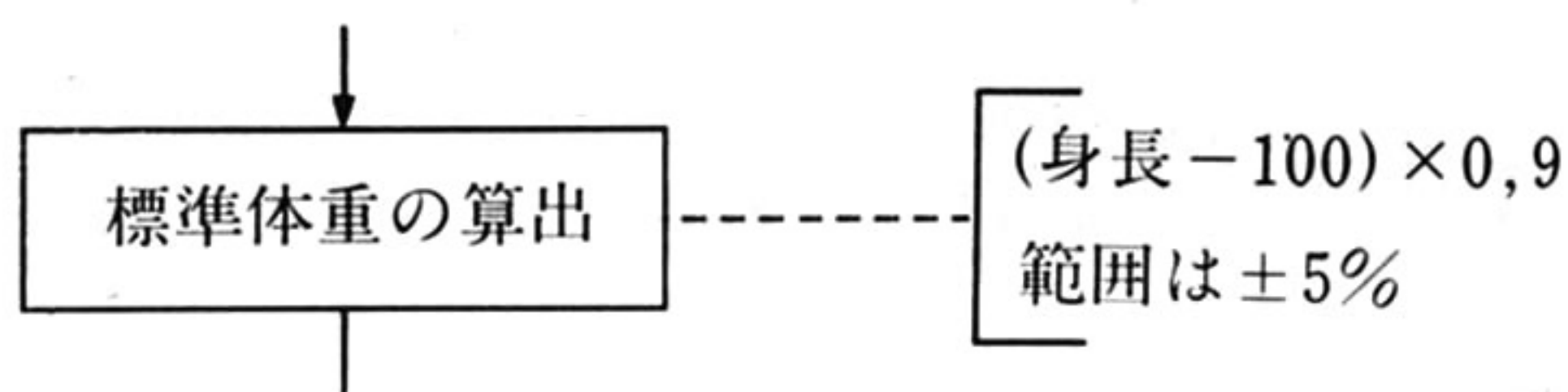
1行改行

⑤

```
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
```

・この三つの行は，③で入力した身長から標準体重を求めています。

・この処理は次のように表します。

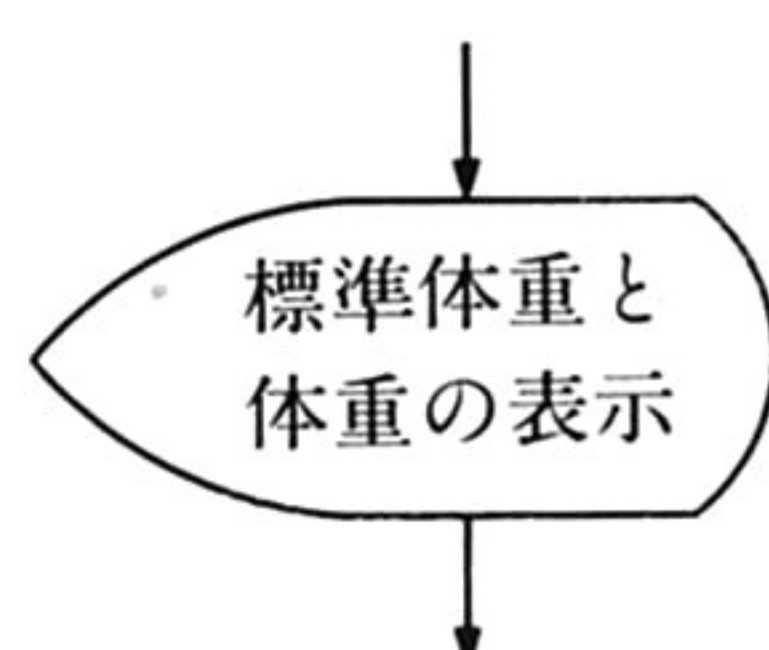
標準体重の算出

(身長－100)×0,9
範囲は±5%

⑥

```
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
```

・この二つの行は，標準体重と③で入力した体重を表示しています。

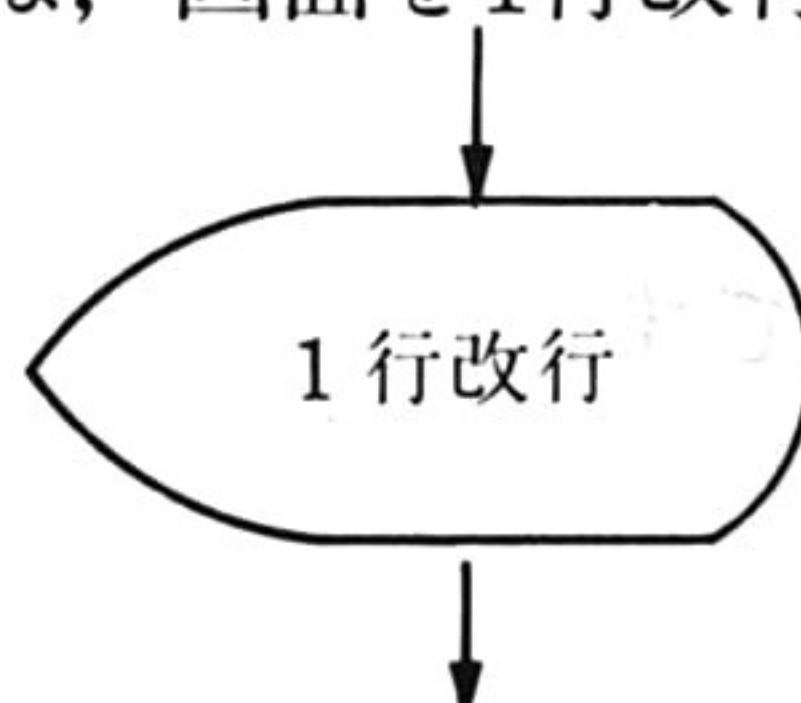
・この処理は次のように表します。

標準体重と
体重の表示

⑦

```
110 PRINT
```

・この行は，画面を1行改行しています。

1行改行

⑧

```
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
```

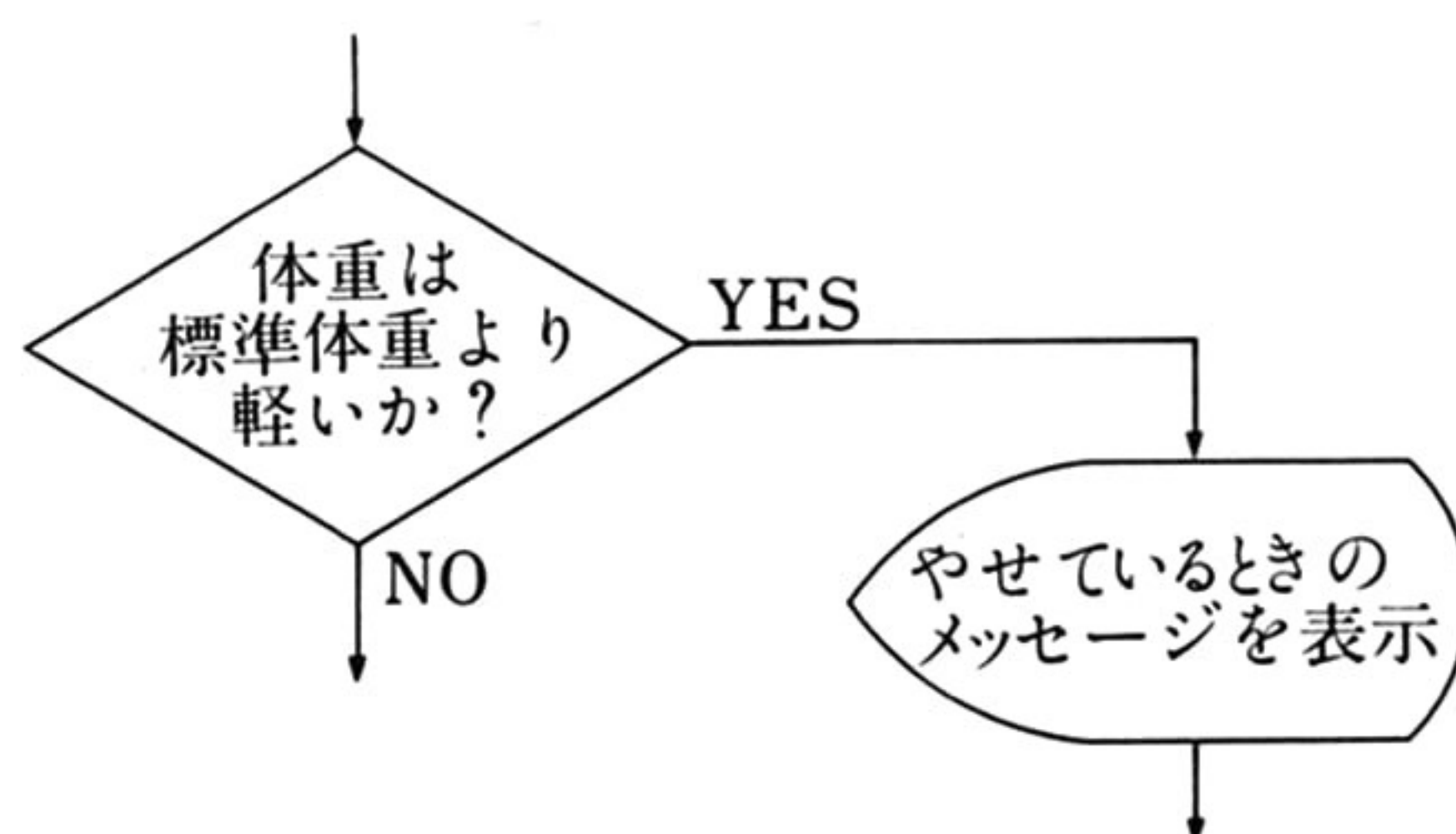
・この行は，標準体重よりやせていた場合にメッセージを表示して分岐する処理を行っています。

・この処理は次のように表します。

体重は
標準体重より
軽いか？

YES

NO

やせているときの
メッセージを表示

⑨

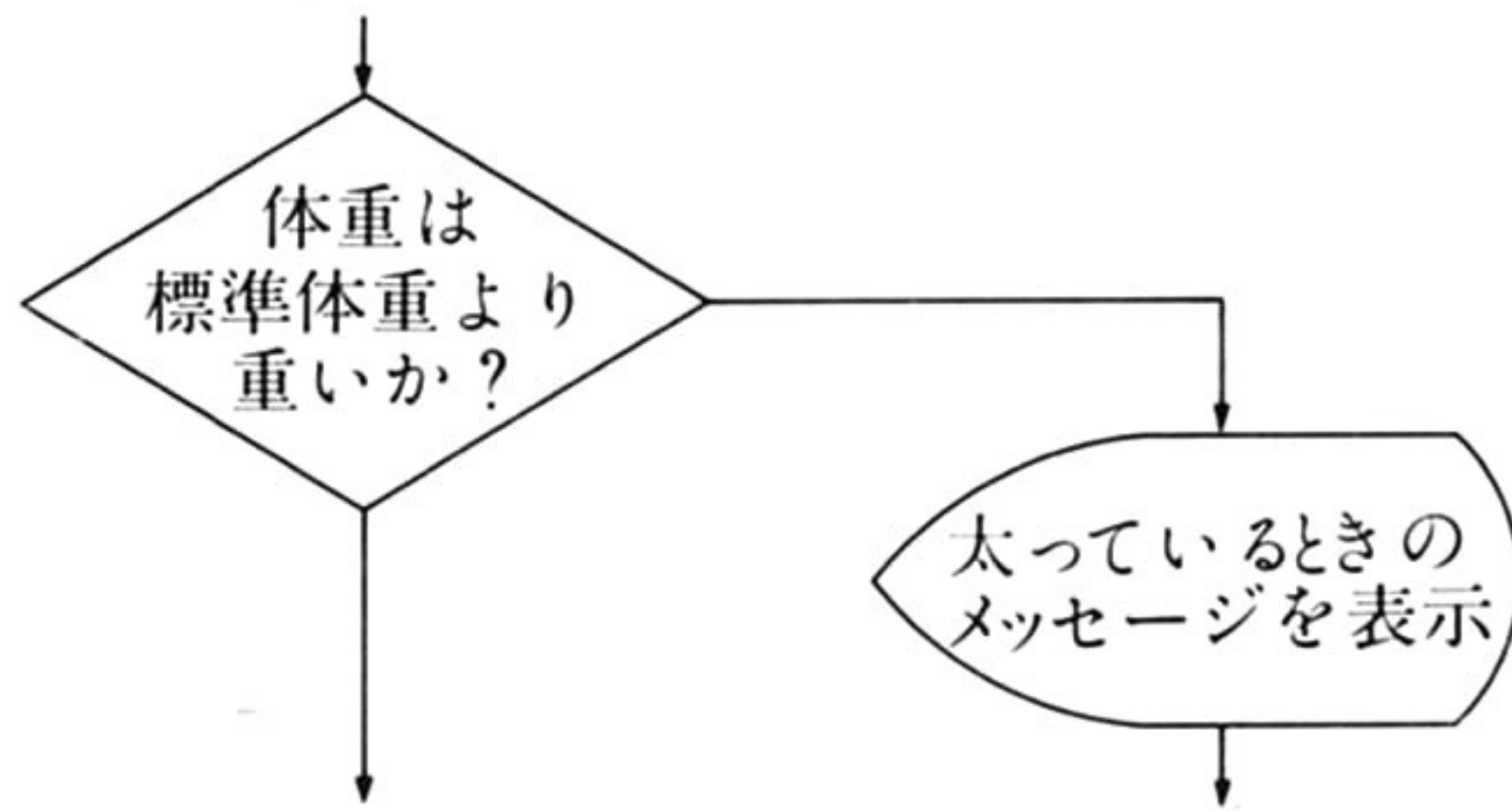
```
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
```

・この行は，標準体重より太っている場合にメッセージを表示して分岐する処理を行っています。

・この処理は次のように表します。

体重は
標準体重より
重いか？

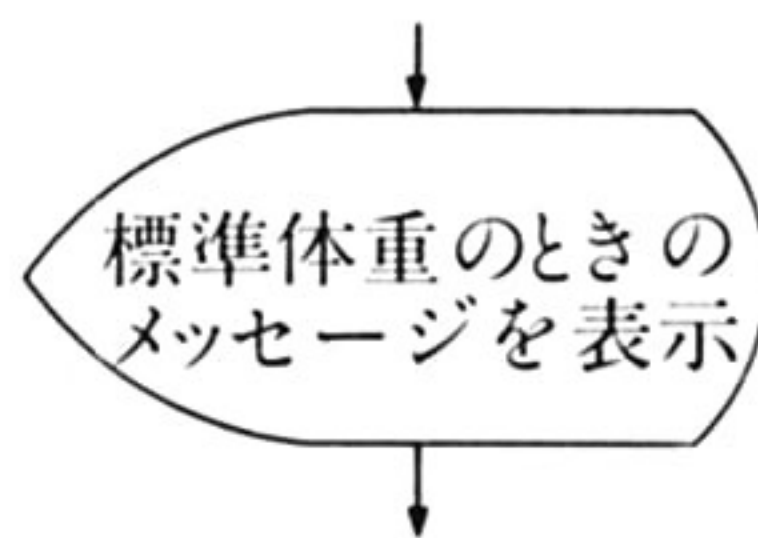
太っているときの
メッセージを表示

⑩

```
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
```

・この行は，標準体重である場合に対応するメッセージを表示しています。

・⑧と⑨の処理でそれぞれの条件に合わないデータ(標準体重のデータ)のときにこの行を実行するため，無条件に処理を行っています。

・この処理は次のように表します。

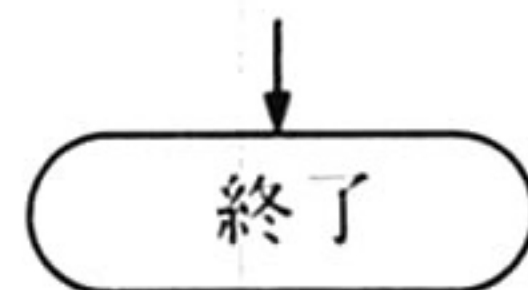
標準体重のときの
メッセージを表示

⑪

```
150 END
```

・この行は，プログラムの終了を行っています。

・⑧で分岐したときも，⑨で分岐したときも，⑩の処理を実行したときも，次にはこの行を実行します。

・この処理は次のように表します。

終了

①～⑪のフローチャート記号をつないで一つのフローチャートにすると，次のようになります。

なお，処理の流れを表す矢印は，線だけで表示することもあります。また矢印は原則として上から下へ，右から左へ書きます。下から上へ，または左から右へ流れるときは，必ず先頭の矢印を付けます。

【 体重比較プログラムのフローチャート 】

開始

画面クリア

データ入力 ― ●身長の入力 ●体重の入力

1行改行

標準体重の算出 ― (身長－100)×0.9 範囲は±0.5%

標準体重と体重の表示

1行改行

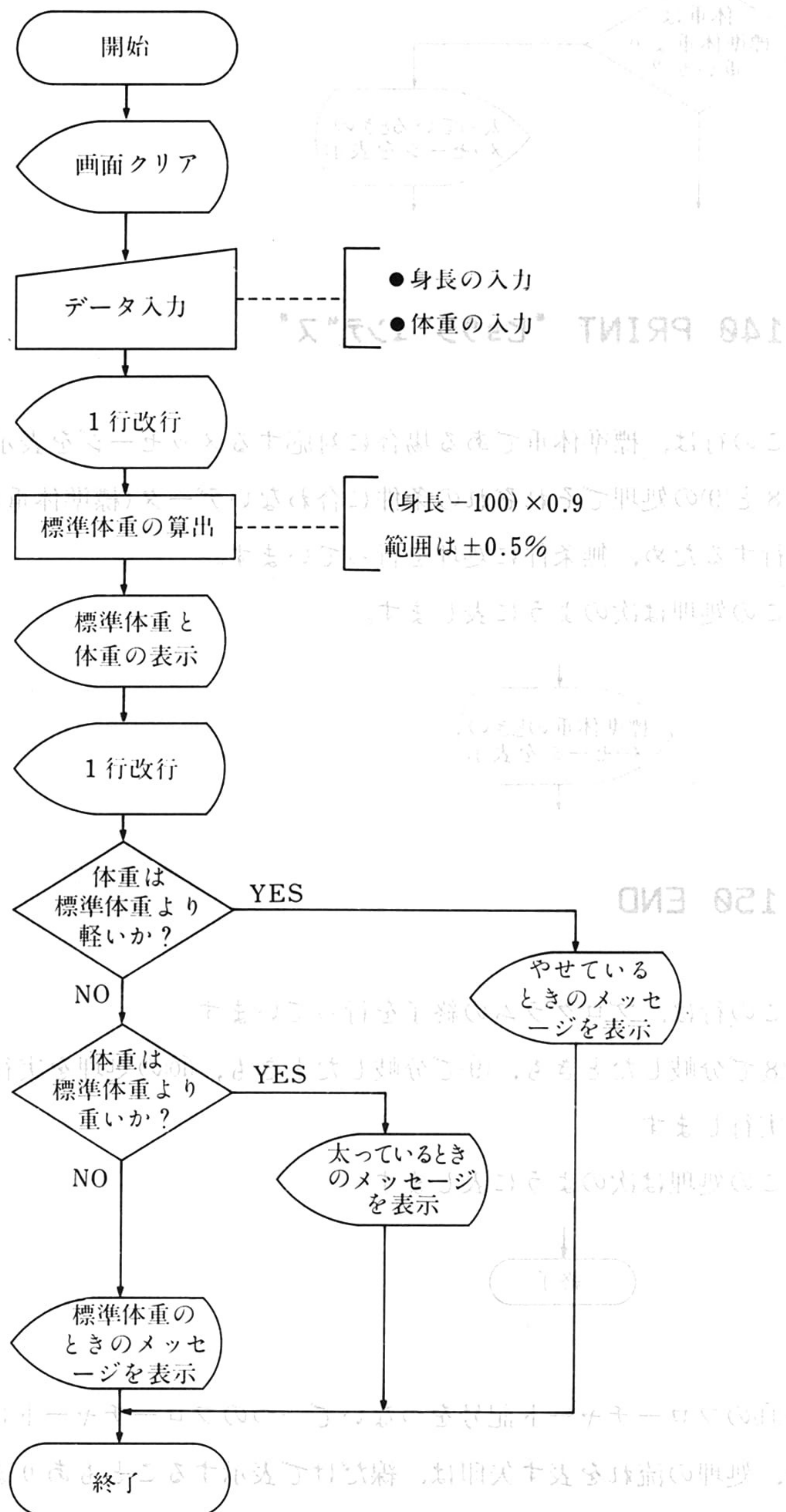
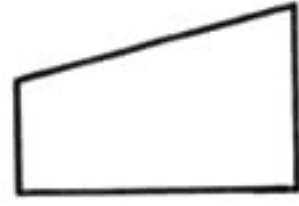
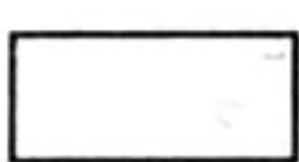
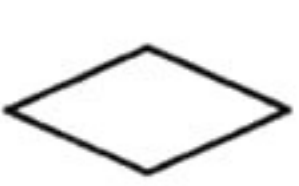
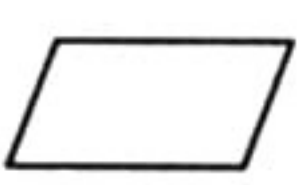
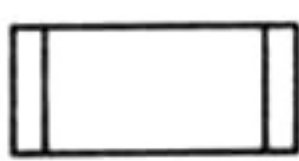
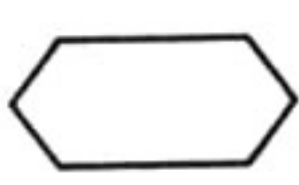
体重は標準体重より軽いか？ YES → やせているときのメッセージを表示

NO

体重は標準体重より重いか？ YES → 太っているときのメッセージを表示

NO

標準体重のときのメッセージを表示

終了

2.4.3 フローチャート記号のいろいろ

フローチャート記号は使用基準(JIS規格)が決められており,次に示すように目的によっていろいろな記号を使い分けます。

記号	意　　味	機　能
	処理の開始，中断，終了を表示する。	端子
	キーボード操作をするとき使用する。	キーボード操作
	画面表示をするとき使用する。	表示
	処理の機能を表す。	処理
	いくつかの選択があるとき，一つの選択要素を決める判断に使用する。	判断
	データの入出力を表すとき使用する。	入出力
	フロッピーディスクに対してのデータの入出力を表すときに使用する。	フロッピーディスク
	処理結果をプリンタに出力するときに使用する。	プリンタ出力
	サブルーチンなど，別の場所で定義した処理ルーチンを表すとき使用する。	定義済み処理
	ルーチンの初期値設定，スイッチの設定に使用する。	準備
	流れ図のつなぎを表示する。	結合子
	処理の内容に説明，注意を与えるとき使用する。	コメント

そのほか，磁気テープや磁気ディスクに対する入出力記号，オンライン処理やオフライン処理などを表す記号など，さまざまな記号があります。

2.5　プログラムの修正方法

プログラムに誤りがあったり，プログラムの拡張などで命令の追加があった場合，プログラムの修正が必要になります。

命令の記述に誤りがあったときや，命令の簡単な追加だったりしたときは，直接プログラムを修正できますが，プログラムの流れそのものを変更するなどの大きな修正があるときは，フローチャートを手直しした上でプログラムを修正します。

2.5.1　体重比較プログラムの修正

(1) プログラムの修正の条件

今まで作成してきたプログラムは，一件分のデータしか処理できません。そこで，複数のデータを処理できるように，今メモリーに記憶しているプログラムを修正します。

プログラムを修正する条件は次のとおりです。

【 修正する条件 】

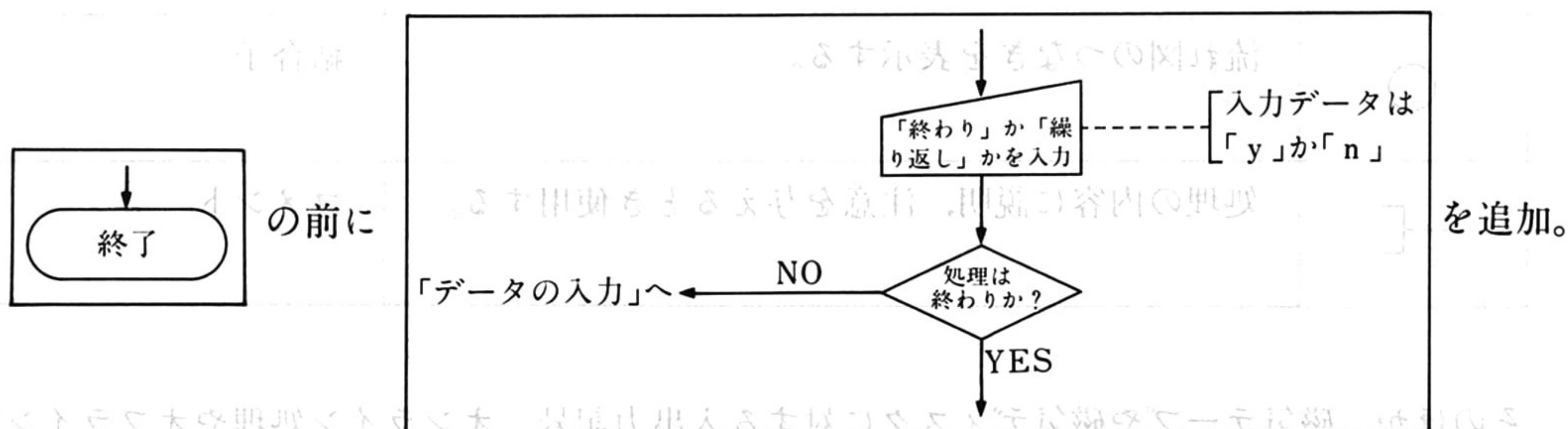
- 入力データが終了するまで処理を繰り返すようにします。
- 処理は最低1回は実行できるようにします。
- 入力データの終了は，一件ごとに確認します。
- データ終了のときはキーボードから「y⏎」を入力し，終了でなければ「n⏎」を入力します。
- 「y⏎」が入力されたらプログラムの実行を終了し，「y⏎」以外が入力されたら再び身長，体重の入力に戻ります。

(2)　フローチャートの修正

フローチャートは，次のように修正します。

・ 終了 の前に

「終わり」か「繰り返し」かを入力 - - - 入力データは「y」か「n」
処理は終わりか？
NO → 「データの入力」へ
YES ↓

を追加。

すると，フローチャートは次のように変わります。

【 修正後のフローチャート 】

開始

画面クリア

データ入力

●身長の入力

●体重の入力

1 行改行

標準体重の算出

(身長－100)×0.9

範囲は±0.5%

標準体重と体重の表示

1 行改行

体重は標準体重より軽いか？

YES

やせているときのメッセージを表示

NO

体重は標準体重より重いか？

YES

太っているときのメッセージを表示

NO

標準体重のときのメッセージを表示

「終わり」か「繰り返し」かを入力

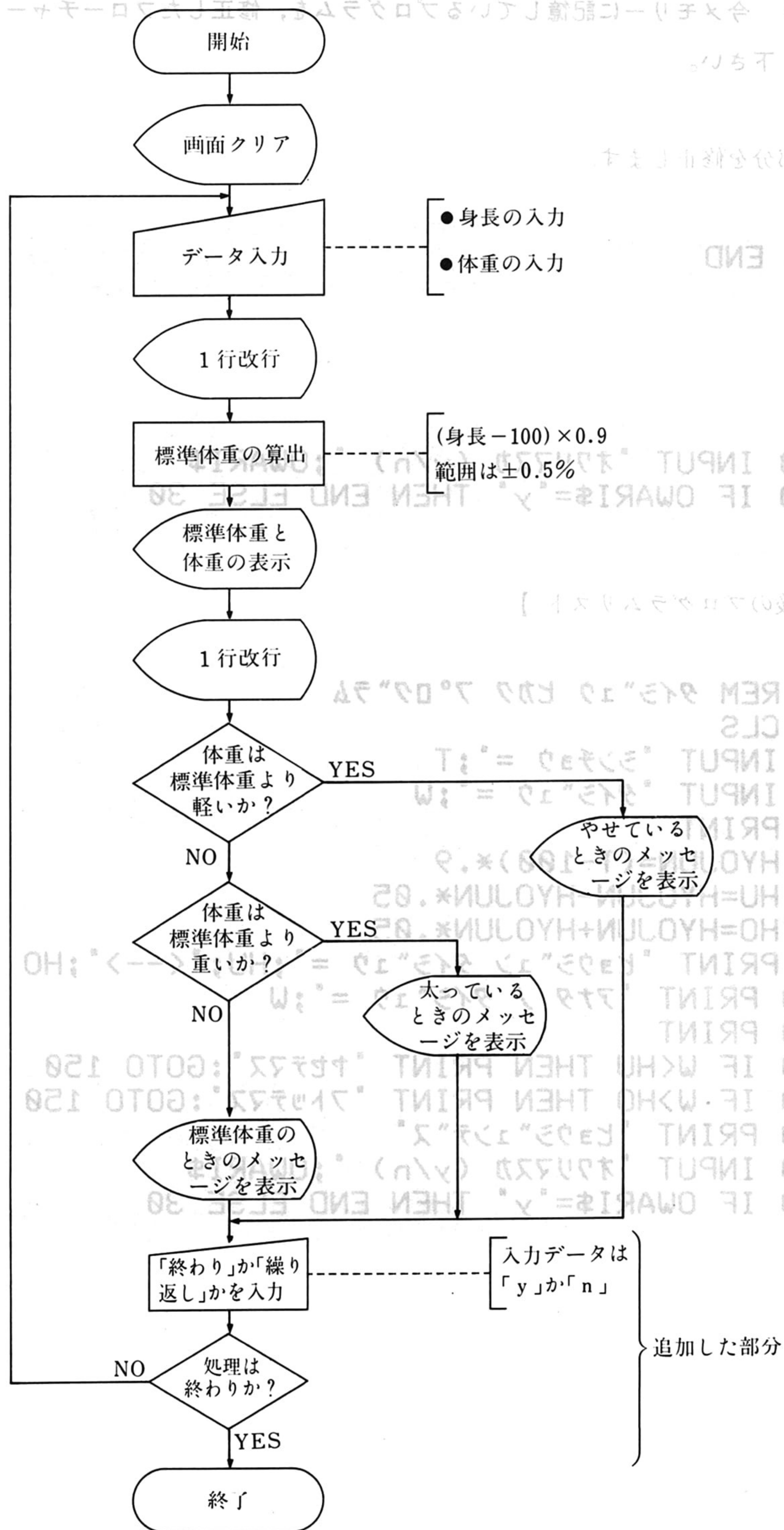
入力データは「y」か「n」

処理は終わりか？

NO

YES

追加した部分

終了

（3） リストの修正

修正したフローチャートに従って，メモリーのプログラムを修正します。

設問 13 今メモリーに記憶しているプログラムを，修正したフローチャートに従って修正して下さい。

次の部分を修正します。

```
150 END
```

⇩

```
150 INPUT "ｵﾜﾘﾏｽｶ (y/n) ";OWARI$
160 IF OWARI$="y" THEN END ELSE 30
```

【 修正後のプログラムリスト 】

```
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 INPUT "ｵﾜﾘﾏｽｶ (y/n) ";OWARI$
160 IF OWARI$="y" THEN END ELSE 30
```

設問 14　今修正したプログラムを実行して下さい。

【 実行結果例 】

```
シンチョウ =? 163
タイジ゛ュウ =? 56

ヒョウジ゛ュン タイジ゛ュウ = 53.865 <--> 59.535
アナタ ノ タイジ゛ュウ = 56

ヒョウジ゛ュンデ゛ス
オワリマスカ (y/n) ? n
シンチョウ =? 170
タイジ゛ュウ =? 72

ヒョウジ゛ュン タイジ゛ュウ = 59.85 <--> 66.15
アナタ ノ タイジ゛ュウ = 72

フトッテマス
オワリマスカ (y/n) ? y
Ok
```

設問 15　今メモリーに記憶しているプログラムを「TAIJU.2」という名前でドライブ2のフロッピーディスクへ保存して下さい。

【 実行結果例 】

```
save "2:TAIJU.2"
Ok
```

2.5.2　プログラムの印字

今までは，プログラムの内容を見るためにプログラムリストを画面に表示していましたが，プログラムリストをプリンタに印字することもできます。プログラムリストをプリンタに印字するには「LLIST」コマンドを使います。LLISTコマンドの一般形式と書き方例は次のとおりです。

(LLISTコマンドの説明)

<一般形式>

LLIST　開始行番号－終了行番号

・メモリーに記憶しているプログラムをプリンタに印字します。

・行番号を一つだけ指定すると，その行だけを印字します。

・開始行番号を省略すると，プログラムの先頭の行から終了行番号として指定した行までを印字します。

・終了行番号を省略すると，開始行番号として指定した行からプログラムの最後の行までを印字します。

・開始行番号と終了行番号の両方を省略すると，プログラムの先頭の行から最後の行まですべて印字します。

<書き方例>

① LLIST　20－80	メモリーに記憶しているプログラムの行番号20から行番号80までの行をプリンタに印字しています。
② LLIST　－60	プログラムの先頭の行から行番号60までの行をプリンタに印字しています。
③ LLIST　50－	行番号50からプログラムの最後の行までを印字しています。
④ LLIST　30	行番号30の行だけを印字しています。
⑤ LLIST	メモリーに記憶されているプログラムのすべての行を印字しています。

設問 16　今メモリーに記憶しているプログラムをすべてプリンタに印字して下さい。

【 実行結果例 】

(画面)

```
llist
Ok
```

(プリンタ)

```
10 REM ﾀｲｼﾞｭｳ ﾋｶｸ ﾌﾟﾛｸﾞﾗﾑ
20 CLS
30 INPUT "ｼﾝﾁｮｳ =";T
40 INPUT "ﾀｲｼﾞｭｳ =";W
50 PRINT
60 HYOJUN=(T-100)*.9
70 HU=HYOJUN-HYOJUN*.05
80 HO=HYOJUN+HYOJUN*.05
90 PRINT "ﾋｮｳｼﾞｭﾝ ﾀｲｼﾞｭｳ =";HU;"<-->";HO
100 PRINT "ｱﾅﾀ ﾉ ﾀｲｼﾞｭｳ =";W
110 PRINT
120 IF W<HU THEN PRINT "ﾔｾﾃﾏｽ":GOTO 150
130 IF W>HO THEN PRINT "ﾌﾄｯﾃﾏｽ":GOTO 150
140 PRINT "ﾋｮｳｼﾞｭﾝﾃﾞｽ"
150 INPUT "ｵﾜﾘﾏｽｶ (y/n) ";OWARI$
160 IF OWARI$="y" THEN END ELSE 30
```

3章　配列を使ったプログラム

異なる変数に対して同じ処理を繰り返し行う場合や，多くのデータをメモリーに貯えておかなくてはならない場合などがしばしばあります。そのとき，その都度変数を一つ一つ使っていたのでは，使用する変数の量もプログラムの行数(ステップ数)も膨大なものになります。また，使っている変数も何のために使っているのかその目的さえもわからなくなってしまいます。

そういったプログラム上の無駄や紛らわしさをなくすために，「配列」という便利なものがあります。

この章では，配列の使い方を中心に学習して行きます。

3.1　配列とは

配列とは，異なる番号を持った同じ名前の変数をいくつも定義しておくもので，一つ一つの変数の区別は番号で行われます。その定義した変数全体のことを「配列」と呼び，配列に付ける名前を「配列名」，配列を構成している変数一つ一つを「配列要素」，配列要素を区別している番号を「添字」と言います。

また，身近なものにたとえると，「あるアパート(配列)があってそれに名前(配列名)が付いています。そして一部屋一部屋(配列要素)には部屋番号(添字)が付いています。」という具合になります。

3.2　プログラム例(2) —— 宣伝効果を調査するプログラム

配列を使った例として，前月の宣伝費と当月の売上高から宣伝の効果を調査するプログラムを作成することにしましょう。

3.2.1　プログラムに必要な条件

このプログラムを作るために必要な条件は次のとおりです。

【 条件 】

・入力がすべて終了するまでデータは配列に貯えておきます。

・データの入力が終了したら，当月の売上高に対する前月の宣伝費の比率を算出します。

・入力するデータは，対象となる月，前月の宣伝費，当月の売上高です。

・一年分のデータが入力できるように，データの入力は12回まで繰り返すようにします。

・対象となる月のデータとして「999」が入力されたら，表示の処理を行います。

・宣伝効果を表す比率の算出は次の式で行います。

宣伝費の比率＝前月宣伝費÷当月売上×100

・表示するデータは，対象となる月と宣伝費の比率です。

3.2.2　フローチャートを作る

まず，宣伝効果調査プログラムのフローチャートを作成します。

【 フローチャート例 】

開始

配列の定義 ---- ●対象月 ●前月宣伝費 ●当月売上

画面クリア

カウンタAを初期値クリア

(A) →

対象月の入力 ---- 直接配列に入力

入力データは「999」か　YES → (B)

NO

前月宣伝費の入力 ---- 直接配列に入力

当月売上の入力 ---- 直接配列に入力

カウンタAに1を加算

カウンタAは12を超えたか？　NO → (A)

YES

(B) →

カウンタBを初期値クリア

(C) →

対象月は「999」か？　YES → (D)

NO

宣伝費の比率を算出 ---- 宣伝費の比率＝前月宣伝費／当月売上＊100

対象月と宣伝費の比率を表示

カウンタBに1を加算

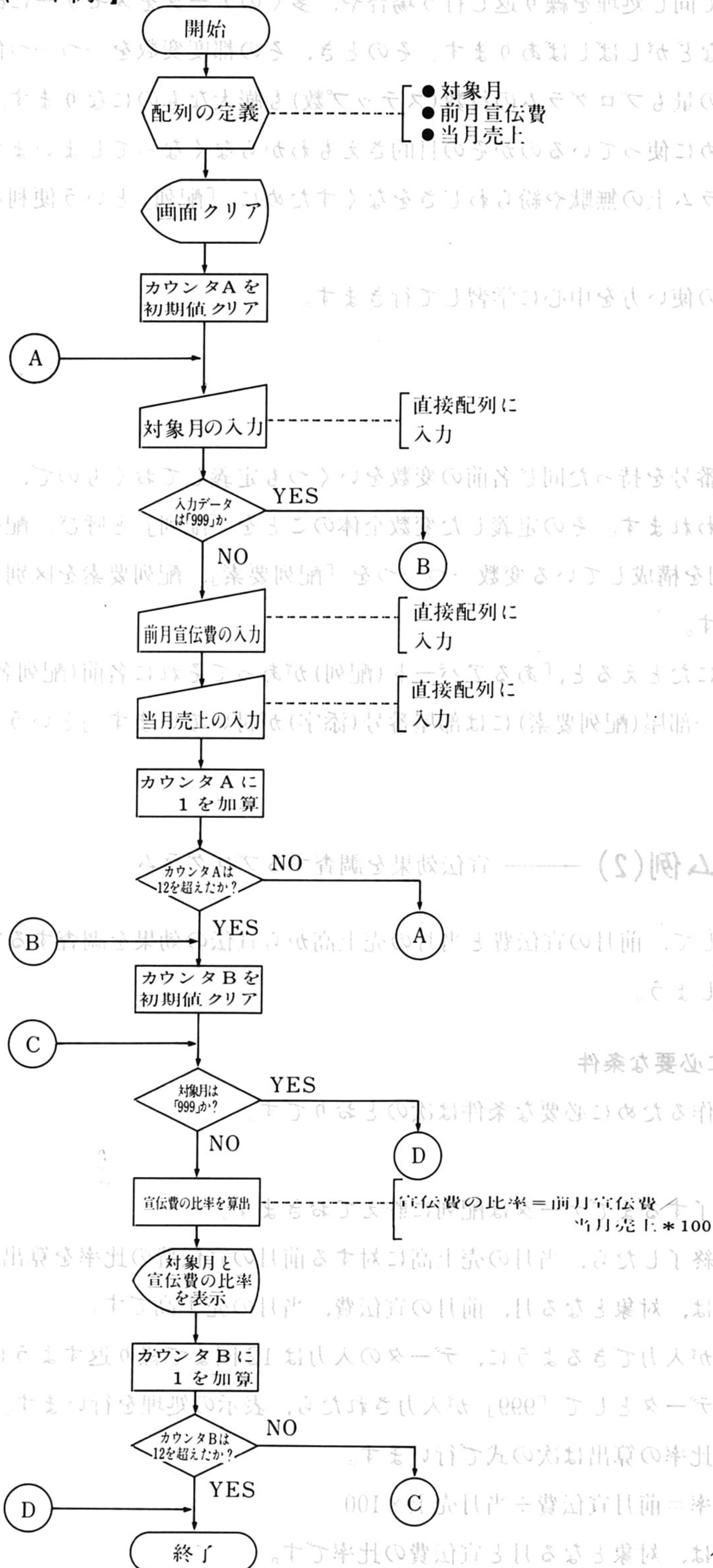
カウンタBは12を超えたか？　NO → (C)

YES

(D) →

終了

入力処理（開始〜カウンタAの判定）

出力処理（カウンタBを初期値クリア〜終了）

3.3 プログラムに必要な命令の説明

前項で作成したフローチャートからBASIC言語でコーディングして行くわけですが，この項で新たに必要となる命令を説明します。

3.3.1 配列に関する説明

（1） 配列宣言の命令

配列を使用する前には，必ず使用する配列の名前と大きさ(配列要素の数)を宣言します。配列の宣言には「DIM」命令を使います。DIM命令の一般形式と書き方例は次のとおりです。

（DIM命令の説明）

<一般形式>

DIM 配列名 1 (添字の最大値)，配列名 2 (添字の最大値)，………

・メモリーに配列名で定義した配列を確保します。

・確保された配列で使用できる配列要素は，0～「添字の最大値」までです。

・配列名は，プログラムの一行の範囲内であればカンマで区切ることによっていくつでも宣言できます。ただし，すべての配列を合わせた大きさはメモリーの大きさを超えてはいけません。

<書き方例>

① DIM A(15)　　数値配列「A」について「A(0)」から「A(15)」までの16個をメモリーに確保します。

② DIM X(12)，Y＄(20)　　数値配列「X」を13個，文字配列「Y＄」を21個メモリーに確保します。

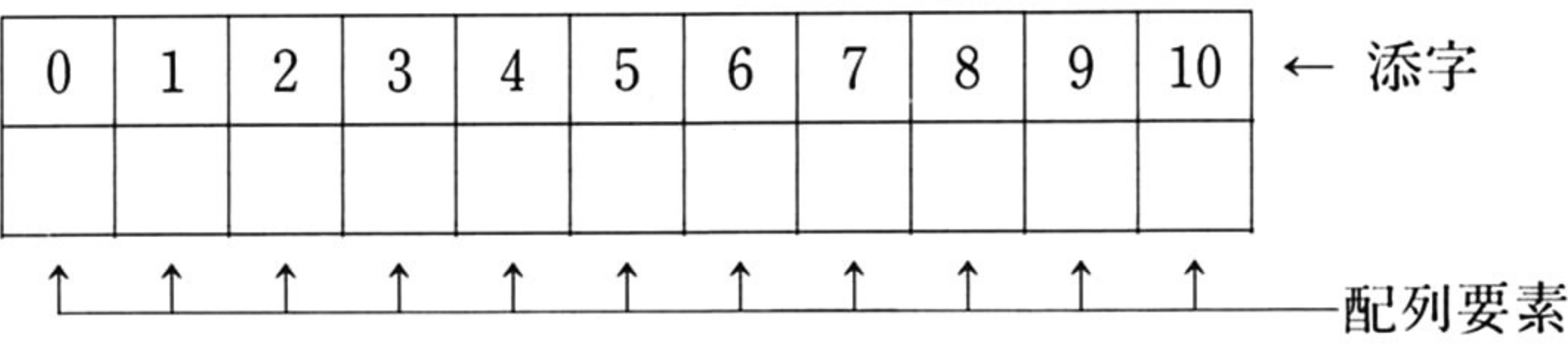
（2） 配列の使い方

配列は添字が付いてはいますが，変数と同じ使い方をします。

たとえば，「DIM NA＄(10)」と書くと，次の図のように11個の連続した記憶場所が確保されます。

配列（この場所の配列名はNA＄）

0	1	2	3	4	5	6	7	8	9	10

← 添字

↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ── 配列要素

たとえば，左から3番目の配列要素を指定するときには「NA＄(2)」，4番目の配列要素を指定するときは「NA＄(3)」のように書きます。

0	1	2	3	4	5	6	7	8	9	10

↑ NA＄(2)　↑ NA＄(3)

また，配列要素にデータを記憶させるときや，配列要素からデータを取り出すときは次のようにします。

＜書き方例＞

① NA＄(2)＝"HAIRETSU"　配列要素NA＄(2)に「HAIRETSU」という文字列を入れて(記憶)います。

② B＄＝NA＄(5)　配列要素NA＄(5)の内容を変数B＄に入れています。

③ PRINT　NA＄(J)　配列要素NA＄(J)の内容を画面に表示します。変数Jの内容によって指定する配列要素の場所が変ります。

（３） 配列の取り消し

プログラムの中で使用して不要になった配列は「ERASE」命令を使って消すことができます。ERASE命令の一般形式と書き方例は次のとおりです。

(ERASE命令の説明)

＜一般形式＞

ERASE　配列名１，配列名２，………

・宣言された配列をメモリーから削除します。

・指定する配列名はDIM命令ですでに宣言されていなければなりません。

＜書き方例＞

① ERASE　A　数値配列Aを削除し，その場所を他の変数などで自由に使えるようにしています。

② ERASE　X，Y＄　数値配列Xと文字配列Y＄を削除しています。

メモリーを節約して使うためにも，不要な配列はERASE命令を使って消去しておきます。

3.3.2 繰り返し処理に関する説明

配列を使うときに添字として変数を使用すれば，同じ処理を繰り返すときに変数の値を変えるだけで異なる配列要素が使えて便利です。同じ処理を繰り返す場合，今まではIF文とGOTO文を使いましたが，この他にも同じ処理を繰り返し行う命令があります。

同じ処理を繰り返すときは，「FOR～NEXT」命令を使います。FOR～NEXT命令の一般形式と書き方例は次のとおりです。

(FOR～NEXT命令の説明)

<一般形式>

```
FOR  変数＝初期値  TO  最終値  STEP  増分
      :
 (一連に繰り返す命令の集まり)
      :
NEXT  変数
```

- FOR命令で指定した変数の値が初期値から始まり，FOR命令とNEXT命令の間の命令を繰り返し実行する度に変数に増分を加え，変数の値が最終値を超えるまで行います。
- 変数には数値変数を指定し，カウンタとして使われます。
- 初期値には，変数の増減の開始の値を指定します。
- 最終値には，変数の増減の終了の値を指定します。
- 増分には，変数の増減のきざみの値を指定します。
- 増分で正の値を指定すれば処理を繰り返す度に変数の値は増え，負の値を指定すれば繰り返す度に減って行きます。
- 増分が「1」のときはSTEP以降を省略することもできます。
- 初期値，最終値，増分の指定は，それぞれ数値定数，数値変数，数式のいずれでもかまいません。

<書き方例>

①
```
FOR  A=1  TO  10  STEP  2
   :
NEXT  A
```

変数Aの値が「1」から始まり「2」ずつ増加させて「10」を超えるまで，つまり5回FORとNEXTの間の命令を繰り返し実行します。

② FOR B=15 TO 1 STEP −3

　　⋮

NEXT B

変数Bの値を「15」から「3」ずつ減少させて「1」より小さくなるまで5回繰り返します。

③ FOR C=0 TO 5

　　⋮

NEXT C

変数Cの値を「0」から「1」ずつ増加させて「5」を超えるまで6回繰り返します。

④ FOR D=X TO Y STEP Z

　　⋮

NEXT D

変数Dの値を「変数Xの値」から「変数Zの値」ずつ増加させて「変数Yの値」を超えるまで繰り返します。

⑤ FOR E=X+2 TO Y+5 STEP Z*3

　　⋮

NEXT E

変数Eの値を「変数Xの内容+2」の値から「変数Zの内容*3」の値ずつ増加させて「変数Yの内容+5」の値が超えるまで繰り返します。

3.4 プログラムのキーインと実行

それでは，作成したフローチャートに従ってBASICの命令でコーディングして行くことにしましょう。「3.2.2　フローチャートを作る」で作成したフローチャートから配列やFOR～NEXT命令などを使ってコーディングすると，プログラムは次のようになります。

【 コーディング例 】

```
10 ' ｾﾝﾃﾞﾝｺｳｶ ﾁｮｳｻ ﾌﾟﾛｸﾞﾗﾑ
20 DIM MONTH(12),CM#(12),URI#(12)
30 CLS
40 FOR I=1 TO 12
50   INPUT "ﾅﾝｶﾞﾂ (END=999) ";MONTH(I)
60   IF MONTH(I)=999 THEN *RITSU
70   INPUT "ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =";CM#(I)
80   INPUT "ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =";URI#(I)
90 NEXT I
100 *RITSU
110 FOR J=1 TO 12
120   IF MONTH(J)=999 THEN *OWARI
130   KOUKA=CM#(J)/URI#(J)*100
140   PRINT MONTH(J);"月",KOUKA;"%"
150 NEXT J
160 *OWARI:END
```

3.4.1 行番号の自動発生

BASICでコーディングされたプログラムは一行ごとに行番号を付けてキーインして行きますが，短いプログラムならともかく長いプログラムになると，一行ごとに行番号までキーインするのは大変です。そんなときにPC-8801mkIIが行番号を自動的に発生させてくれると，プログラムのキーインが非常に楽になります。その行番号を自動的に発生させる機能がPC-8801mkIIには用意されています。

行番号を自動的に発生させるには「AUTO」コマンドを使います。AUTOコマンドの一般形式と書き方例は次のとおりです。

（AUTOコマンドの説明）

＜一般形式＞

AUTO　開始行番号，増分

- 行番号を自動的に発生させます。
- 開始行番号では発生させる最初の行番号を指定します。指定できる範囲は1～65529です。
- 増分では行番号と行番号の間のきざみの値を指定します。
- 増分を省略すると「10」が指定されたときと同じ働きをします。
- 開始行番号と増分の両方を省略すると，「AUTO　10，10」が指定されたときと同じ働きをします。

＜書き方例＞

①	AUTO　100，2	100から順に102，104，106，… という具合に，2きざみで行番号を発生させます。
②	AUTO　200	200から順に210，220，230，… という具合に，10きざみで行番号を発生させます。
③	AUTO	10から順に20，30，40，… という具合に，10きざみで行番号を発生させます。

設問 17　**前ページのコーディング例のプログラムを行番号を自動的に発生させながらキーインして下さい。**

【 実行結果例 】

```
auto
10 ' ｾﾝﾃﾞﾝｺｳｶ ﾁｮｳｻ ﾌﾟﾛｸﾞﾗﾑ
20 DIM MONTH(12),CM#(12),URI#(12)
30 CLS
40 FOR I=1 TO 12
50   INPUT "ﾅﾝｶﾞﾂ (END=999) ";MONTH(I)
60   IF MONTH(I)=999 THEN *RITSU
70   INPUT "ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =";CM#(I)
80   INPUT "ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =";URI#(I)
90 NEXT I
100 *RITSU
110 FOR J=1 TO 12
120   IF MONTH(J)=999 THEN *OWARI
130   KOUKA=CM#(J)/URI#(J)*100
140   PRINT MONTH(J);"月",KOUKA;"%"
150 NEXT J
160 *OWARI:END
170 ■
```

【 実行後の説明 】

・AUTOコマンドの実行は開始行番号と増分を省略し，行番号10から10きざみで増加していくようにしています。

・行番号20で配列 MONTH(12)，CM#(12)，URI#(12)を宣言しています。

・配列 CM#(12)は対象となる月を記憶します。

・配列 URI#(12)は当月売上を記憶します。

・行番号40～90ではデータ入力の処理を行っています。

・行番号100～140ではデータ出力の処理を行っています。

・行番号60では，MONTH(I)に「999」が入力されたときに出力処理へ分岐します。そのときMONTH(I)には「999」のデータが残ります。

・入力処理は，終わりのデータ(「999」)が入力されなければ12回まで繰り返します。

・行番号120では，MONTH(J)が「999」ならば処理を終了処理へ分岐しています。

・出力処理は，終わりのデータ(「999」)がみつからなければ12回まで繰り返します。

コーディングの最後の行のキーインが終わってもAUTOコマンドはまだ働いていて，行番号170の行をキーインできる状態になっています。そこで，AUTOコマンドを解除します。AUTOコマンドを解除するには STOP キーを押します。

設問 18 AUTOコマンドの機能を解除して下さい。

【 実行手順 】

① 行番号170を表示してカーソルが点滅しているところで STOP キーを押します。

② すると，「170」を残してカーソルは次の行に移ります。

【 実行結果例 】

```
150 NEXT J
160 *OWARI:END
170
■
```

【 実行後の説明 】

・プログラムのキーインが終わったらAUTOコマンドを解除しておかないと，「180　LIST」とか「200　RUN」とか，変な行が紛れ込む原因になります。

設問 19　プログラムが正しくキーインされているかどうかプログラムリストを画面に表示して確認して下さい。

【プログラム表示例】

```
list
10 ' ｾﾝﾃﾞﾝｺｳｶ ﾁｮｳｻ ﾌﾟﾛｸﾞﾗﾑ
20 DIM MONTH(12),CM#(12),URI#(12)
30 CLS
40 FOR I=1 TO 12
50   INPUT "ﾅﾝｶﾞﾂ (END=999) ";MONTH(I)
60   IF MONTH(I)=999 THEN *RITSU
70   INPUT "ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =";CM#(I)
80   INPUT "ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =";URI#(I)
90 NEXT I
100 *RITSU
110 FOR J=1 TO 12
120   IF MONTH(J)=999 THEN *OWARI
130   KOUKA=CM#(J)/URI#(J)*100
140   PRINT MONTH(J);"月",KOUKA;"%"
150 NEXT J
160 *OWARI:END
Ok
```

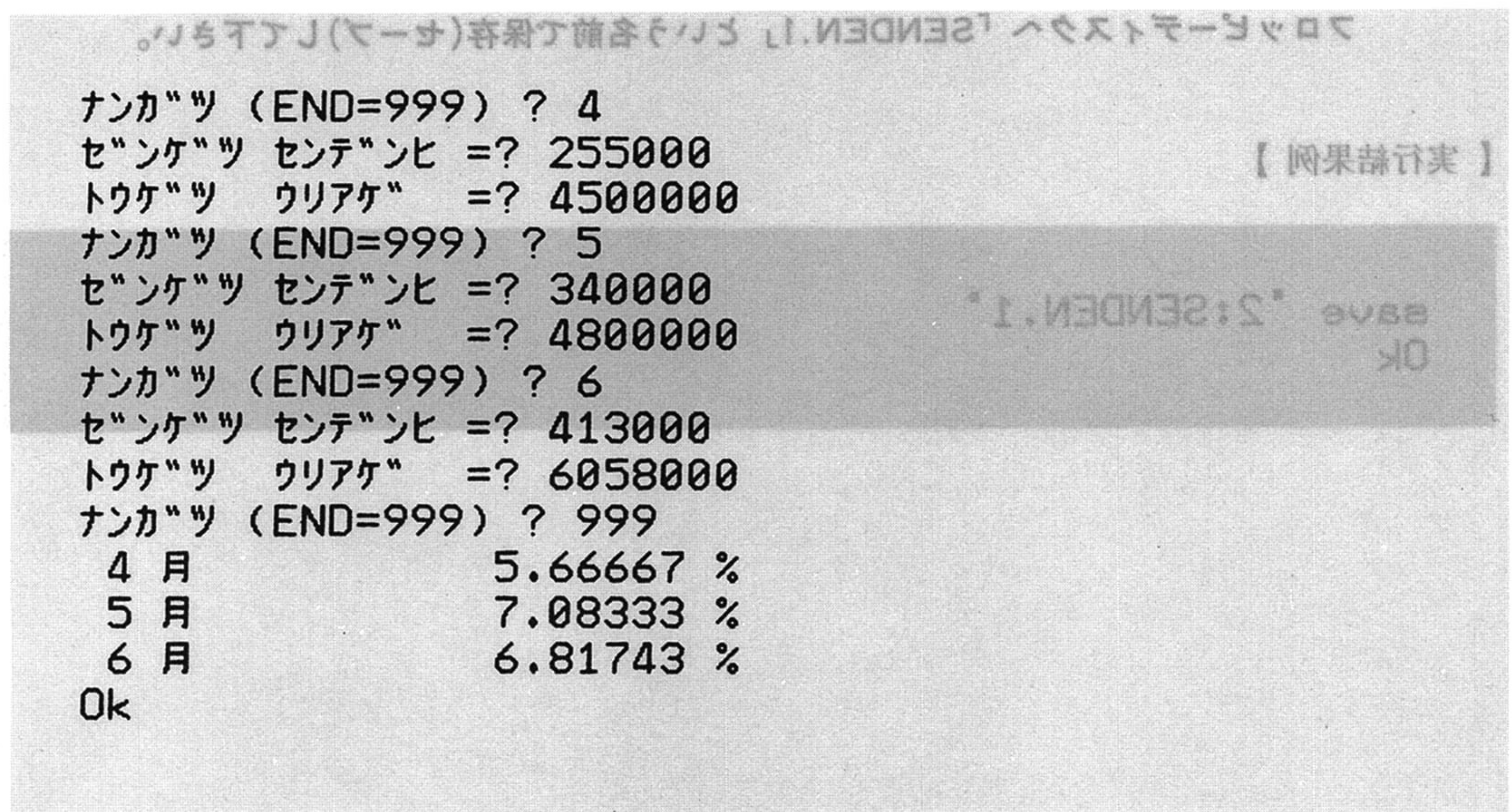
3.4.2 作成したプログラムの実行

設問 20　作成した宣伝効果調査プログラムを，つぎのデータを使って実行して下さい。

（入力するデータ）

対象となる月	前月宣伝費	当月売上
4	255000	4500000
5	340000	4800000
6	413000	6058000
999		

【 実行結果例 】

```
ﾅﾝｶﾞﾂ (END=999) ? 4
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 255000
ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =? 4500000
ﾅﾝｶﾞﾂ (END=999) ? 5
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 340000
ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =? 4800000
ﾅﾝｶﾞﾂ (END=999) ? 6
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 413000
ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =? 6058000
ﾅﾝｶﾞﾂ (END=999) ? 999
 4 月           5.66667 %
 5 月           7.08333 %
 6 月           6.81743 %
Ok
```

【 実行後の説明 】

・対象月に「999」が入力されるかまたは12回になるまで対象月，前月宣伝費，当月売上の入力を繰り返しています。

・対象月と売上に対する宣伝費の比率を結果として画面に出力しています。

・対象月と比率の表示が離れているのは，行番号140のPRINT命令で「月」と変数 KOUKAを続けて表示するときに「,」で区切っているからです。

・対象月に続いて表示される比率が小さいほど宣伝の効果が上がっていると言えるでしょう。

・配列 MONTH，CM#，URI#にはそれぞれつぎのようなデータが記憶されます。

	0	1	2	3	4	………	12 ←添字
MONTH	未使用	4	5	6	999		

	0	1	2	3	4	………	12
CM#	未使用	255000	340000	413000			

	0	1	2	3	4	………	12
URI#	未使用	4500000	4800000	6058000			

設問 21　プログラムの実行が正しければ，今メモリーに記憶しているプログラムをドライブ2のフロッピーディスクへ「SENDEN.1」という名前で保存(セーブ)して下さい。

【 実行結果例 】

```
save "2:SENDEN.1"
Ok
```

3.5　結果をプリンタに出力する

今までは，処理した結果をすべて画面に出力していました。しかしそれでは，電源を切ると画面に表示していた文字は消えてしまって，記録として残りません。また，表示した結果は画面でしか見ることができないため，他の場所で作業する場合にはメモ用紙などに転記しなければならなくなり大変不便です。そこで，処理した結果をプリンタに印字することが必要になってきます。

プリンタに結果を印字するには「LPRINT」命令を使います。LPRINT命令の一般形式と書き方例は次のとおりです。

(LPRINT命令の説明)

＜一般形式＞

LPRINT　式または文字式

・プリンタに文字や数値を印字します。

・式または文字式を省略すると，1行だけ改行します。

＜書き方例＞

①	LPRINT　3+5	プリンタに「3+5」の演算結果「8」を印字します。
②	LPRINT　HENSU	プリンタに変数 HENSUの内容を印字します。
③	LPRINT　”テキスト”	プリンタに「テキスト」という文字列を印字します。
④	LPRINT　MOJI＄	プリンタに変数 MOJI＄の内容を印字します。
⑤	LPRINT	プリンタにセットしている紙を1行だけ改行します。

それでは，画面に表示していた結果をプリンタへ印字するようにプログラムを修正しましょう。

3.5.1　プログラム修正の条件

宣伝効果調査プログラムを修正する条件を次に示します。

【 修正する条件 】

・今まで画面に表示していた対象となる月と当月売上に対する前月宣伝費の比率をプリンタに印字するようにします。

・入力処理で画面のメッセージが見やすくなるように，データ1件分の入力の後，1行改行するようにします。

3.5.2 フローチャートの修正

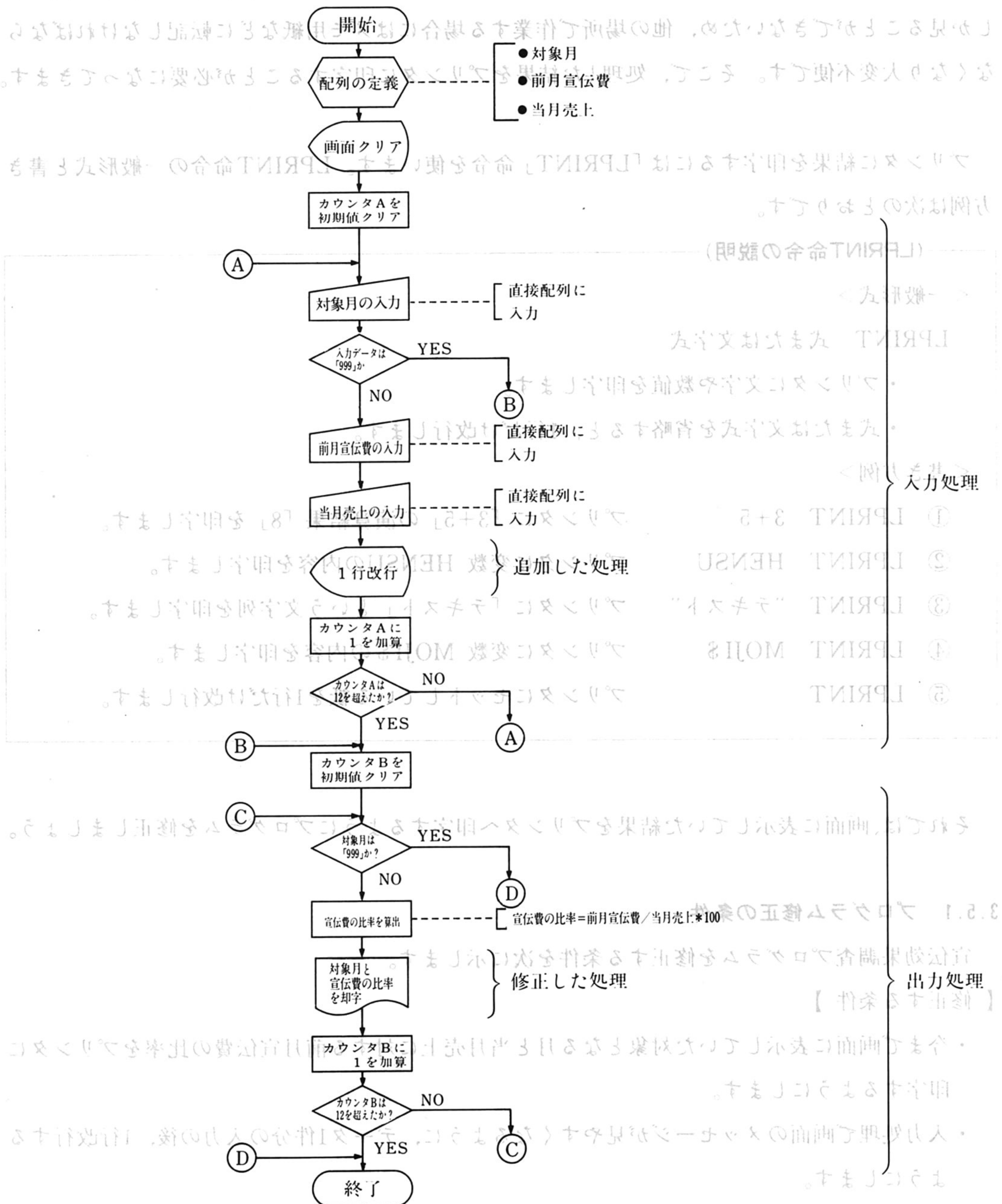
フローチャートを次のように修正します。

【 修正後のフローチャート 】

開始
配列の定義 ●対象月 ●前月宣伝費 ●当月売上
画面クリア
カウンタAを初期値クリア
A
対象月の入力 直接配列に入力
入力データは「999」か YES → B
NO
前月宣伝費の入力 直接配列に入力
当月売上の入力 直接配列に入力
1行改行 追加した処理
カウンタAに1を加算
カウンタAは12を超えたか？ NO → A
YES
B
カウンタBを初期値クリア
C
対象月は「999」か？ YES → D
NO
宣伝費の比率を算出 宣伝費の比率＝前月宣伝費／当月売上＊100
対象月と宣伝費の比率を印字 修正した処理
カウンタBに1を加算
カウンタBは12を超えたか？ NO → C
YES
D
終了

入力処理

出力処理

【 フローチャートの説明】

・「当月売上の入力」の後に「1行改行」の表示処理が追加されています。

・「対象月と宣伝費の比率を表示」の表示処理が，「対象月と宣伝費の比率を印字」の印字処理に修正されています。

・このフローチャートと「3.2.2　フローチャートを作る」のフローチャートとを見比べてみて下さい。

3.5.3　プログラムの修正

修正したフローチャートに従って，今メモリーに記憶しているプログラムを修正します。

設問 22　今メモリーに記憶しているプログラムをすべて画面に表示して下さい。

【 プログラム表示例 】

```
10 ' ｾﾝﾃﾞﾝｺｳｶ ﾁｮｳｻ ﾌﾟﾛｸﾞﾗﾑ
20 DIM MONTH(12),CM#(12),URI#(12)
30 CLS
40 FOR I=1 TO 12
50   INPUT "ﾅﾝｶﾞﾂ (END=999) ";MONTH(I)
60   IF MONTH(I)=999 THEN *RITSU
70   INPUT "ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =";CM#(I)
80   INPUT "ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =";URI#(I)
90 NEXT I
100 *RITSU
110 FOR J=1 TO 12
120   IF MONTH(J)=999 THEN *OWARI
130   KOUKA=CM#(J)/URI#(J)*100
140   PRINT MONTH(J);"月",KOUKA;"%"
150 NEXT J
160 *OWARI:END
```

設問 23 今画面に表示したプログラムを修正したフローチャートに従って修正して下さい。

【プログラム修正例】

```
10 ' センデ゛ンコウカ チョウサ フ゜ログ゛ラム
20 DIM MONTH(12),CM#(12),URI#(12)
30 CLS
40 FOR I=1 TO 12
50   INPUT "ナンカ゛ツ (END=999) ";MONTH(I)
60   IF MONTH(I)=999 THEN *RITSU
70   INPUT "セ゛ンケ゛ツ センデ゛ンヒ =";CM#(I)
80   INPUT "トウケ゛ツ  ウリアケ゛  =";URI#(I)
85   PRINT
90 NEXT I
100 *RITSU
110 FOR J=1 TO 12
120   IF MONTH(J)=999 THEN *OWARI
130   KOUKA=CM#(J)/URI#(J)*100
140   LPRINT MONTH(J);"月",KOUKA;"%"
150 NEXT J
160 *OWARI:END
```

【修正後の説明】

- 行番号85は，入力のときのメッセージが見やすくなるように，1件分のデータを入力した後に1行改行しています。
- 行番号140は，処理した結果をプリンタに印字しています。
- その他の行は，修正前と変わっていません。

3.5.4 修正したプログラムの実行

設問 24 今メモリーに記憶しているプログラムを次のデータで実行して下さい。

（入力するデータ）

対象となる月	前月宣伝費	当月売上
4	255000	4500000
5	340000	4800000
6	41300	6058000
999		

【 実行結果例 】

（画面）

```
ﾅﾝｶﾞﾂ (END=999) ? 4
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 255000
ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =? 4500000
ﾅﾝｶﾞﾂ (END=999) ? 5
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 340000
ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =? 4800000
ﾅﾝｶﾞﾂ (END=999) ? 6
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 413000
ﾄｳｹﾞﾂ  ｳﾘｱｹﾞ  =? 6058000
ﾅﾝｶﾞﾂ (END=999) ? 999
Ok
```

（プリンタ）

```
4 月             5.66667 %
5 月             7.08333 %
6 月             6.81743 %
```

設問 25　プログラムの実行が正しければ，メモリーのプログラムをドライブ2のフロッピーディスクへ「SENDEN.2」という名前でセーブして下さい。

【 実行結果例 】

```
save "2:SENDEN.2"
Ok
```

【実行結果例】

（画面）

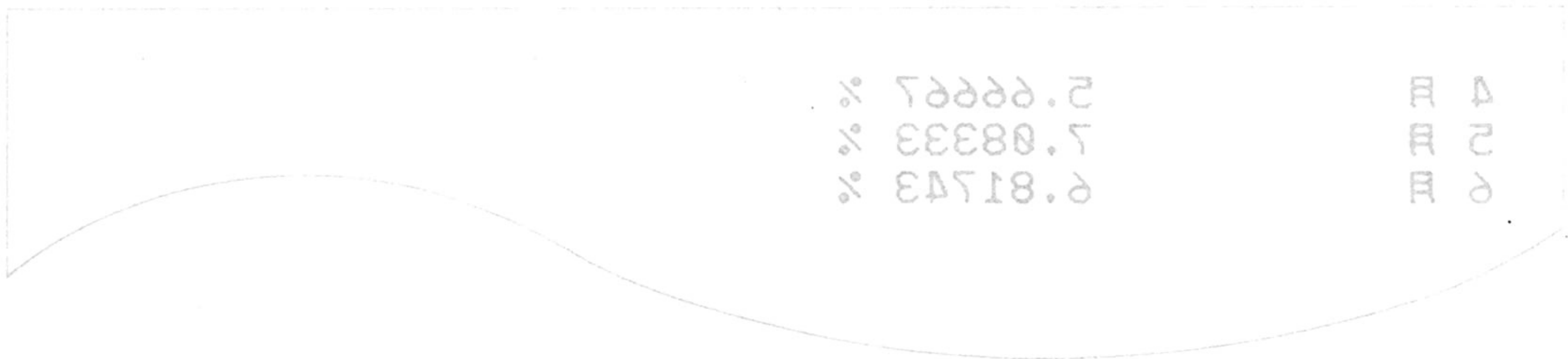
```
ﾅﾝｶﾞﾂ (END=999) ? 4
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 255000
ﾄｳｹﾞﾂ ｳﾘｱｹﾞ  =? 4500000
ﾅﾝｶﾞﾂ (END=999) ? 5
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 340000
ﾄｳｹﾞﾂ ｳﾘｱｹﾞ  =? 4800000
ﾅﾝｶﾞﾂ (END=999) ? 6
ｾﾞﾝｹﾞﾂ ｾﾝﾃﾞﾝﾋ =? 413000
ﾄｳｹﾞﾂ ｳﾘｱｹﾞ  =? 6058000
ﾅﾝｶﾞﾂ (END=999) ? 999
Ok
```

（プリンタ）

```
4 月   5.66667 %
5 月   7.08333 %
6 月   6.81743 %
```

練習 25　プログラムの実行が正しければ，メモリーのプログラムをドライブ2のフロッピーディスクへ「SENDEN.2」という名前でセーブしてください。

【実行結果例】

```
save "2:SENDEN.2"
Ok
```

4章　基本的な処理のテクニック(1)

前章までは条件分岐や配列処理などについて学習しましたが，この章では，今までに学習した命令を利用した，プログラム上の基本的な処理のテクニックを中心に学習して行くことにします。

4.1　プログラム例(3)――――成績処理プログラム

ここでは，国語，数学，英語の3教科のテストの成績を入力し，総合点の高い者から順に表示するプログラムを作成します。

4.1.1　プログラムに必要な条件

プログラムを作成する条件は次のとおりです。

【 条件 】

- データを入力した後，総合点の高い順に並べ替え，画面に表示します。
- 入力するデータは，名前，国語の点数，数学の点数，英語の点数で，それぞれ配列に入力します。
- データは最大10件まで入力できるようにし，10件のデータを入力するか，名前の入力時に「E」または「e」が入力されたら，並べ替えの処理を行います。
- 名前は最大10文字までの入力とします。
- 3教科の点数はそれぞれ0～100の範囲で入力します。
- 配列に入力された3教科の点数を入力したら加算して総合点とし，配列に記憶させます。
- 入力処理の画面サイズは40文字(横)×20行(縦)です。
- 並べ替えの処理の後，画面に見出しと明細を表示します。
- 出力処理の画面サイズは80×20です。
- 画面表示の設計(レイアウト，Layout)は次のとおりです。

ｼﾞｭﾝｲ　ﾅﾏｴ　ｺｸｺﾞ　ｽｳｶﾞｸ　ｴｲｺﾞ　ｿｳｺﾞｳ

4.1.2　フローチャートを作る

まず，フローチャートを作ります。

【 フローチャート例 】

開　始

配列の宣言

ファンクションキーの内容の表示を消す

画面サイズの変更 ---- 横40文字 縦20行

カウンタAを初期値クリア

準備処理

名前の入力

入力データはエンド・データか？　YES / NO

国語の点数を入力

数学の点数を入力

英語の点数を入力

総合点の算出

カウンタAに1を加算

カウンタAは10を超えたか？　NO / YES

入力処理

A

A

カウンタAから
1を減算

カウンタBを
初期値クリア

ワーク変数1に
カウンタBに代入

カウンタCを
初期値クリア

配列(総合点)
を比較

総合点(ワーク1)＜総合点(カウンタC)

ワーク1に
カウンタCを代入

総合点(ワーク1)
≧総合点(カウンタC)

カウンタCに
1を加算

カウンタCは
カウンタAの値
を超えたか?

NO

YES

配列中のデータ
を交換する

添字ワーク1と添字カウンタBの
配列要素の内容と交換

カウンタBに
1を加算

カウンタBは
カウンタAの値－1
を超えたか?

NO

YES

B

並べ替え処理

B

画面サイズの表示 ―― 横80文字 縦20行

見出しの表示

カウンタDを初期値クリア

明細(配列の内容)を表示

カウンタDに1を加算

カウンタDはカウンタAの値を超えたか? NO / YES

ファンクションキーの内容を表示する

終　了

出力処理

後処理

(注) ［結合子の記号］はページをまたがってフローチャートを書いたときに使う結合子です。

【 説明 】

- カウンタAはデータ入力時の添字に使われます。
- カウンタB, カウンタC, ワーク1は並べ替え処理の添字に使われます。
- カウンタDはデータ出力時の添字に使われます。
- カウンタA～Dの初期値はいずれも「1」です。
- 入力処理の後カウンタAから1を減算しているのは，入力処理を終わったときに「1」だけ余分にカウントされているためです。以後カウンタAの値をデータの記録されている配列の添字の最大値としています。

4.1.3 データの並べ替え(ソート)

(1) ソートとは

成績処理プログラムで必要となる並べ替えの処理はソート(SORT)と呼ばれ，実務の上では頻繁に使われる最も重要な処理の一つです。

ソートとは，配列やフロッピーディスク上の一連のデータをある特定の指定項目について，昇順または降順に並べ替える処理のことを言い，分類あるいは整列とも呼ばれます。また，ソートの基準となる特定の指定項目のことを「ソートキー」と呼んでいます。

ソートを行うことによって，さまざまな処理が実行しやすくなります。

(2) 昇順(アセンディング)と降順(ディセンディング)

ソートした結果には二通りのものがあります。一つはソートキーの小さいものから順にソートされたもので，もう一つはソートキーの大きいものから順にソートされたものです。前者を「昇順(アセンディング，Assending)にソートする」，後者を「降順(ディセンディング，Dissending)にソートする」と言います。

昇順のソートと降順のソートは次の例のようになります。

【 ソート例 】

(もとのデータ)

4	5	1	3	2

(昇順にソート)

1	2	3	4	5

(降順にソート)

5	4	3	2	1

〈一口メモ〉

ソートキーと優先順位

ソート(SORT)行うときに，その大小を比較する項目を「ソートキー」と呼びます。

ソートキーが一つのときは，あまり問題はありませんが，例えばクラス別の成績順にソートするというように，ソートキーが二つ以上必要になるときは注意しなければなりません。ソートキーが二つ以上になるときは，ソートキーに優先順位を付け，優先順位の高いものから順に比較します。これを簡単に説明すると次のようになります。

初めに第1キー（優先順位の最も高いキー）について比較を行い，第1キー同士が等しいとき（「＝」のとき）は次に第2キーについて比較を行います。第2キー同士が等しいときは更に第3キーについて比較を行うというように優先順位の高いものから順に比較することによってデータの大小を判定します。

4.1.4 ソートの方法

ソートを行うにはさまざまな方法がありますが，ここでは配列の内容をソートする方法についていくつか説明します。

(1) 逐次決定法

ソートキーが昇順または降順に並ぶように，データの位置を順番に決定して行く方法です。逐次決定法で昇順にソートする例を次に示します。

【 逐次決定法の例 】

①

1	2	3	4	5
4	5	1	3	2

⇨

1	2	3	4	5
1	5	4	3	2

← 添字

決定

比較

交換

・最も小さな値の「1」と「4」を交換します。これで「1」の位置が決定したため，以後「1」は比較の対象にはしません。

⇩

②

1	2	3	4	5
1	5	4	3	2

⇨

1	2	3	4	5
1	2	4	3	5

決定

比較

交換

・「2」と「5」を交換します。

⇩

③

1	2	3	4	5
1	2	4	3	5

⇨

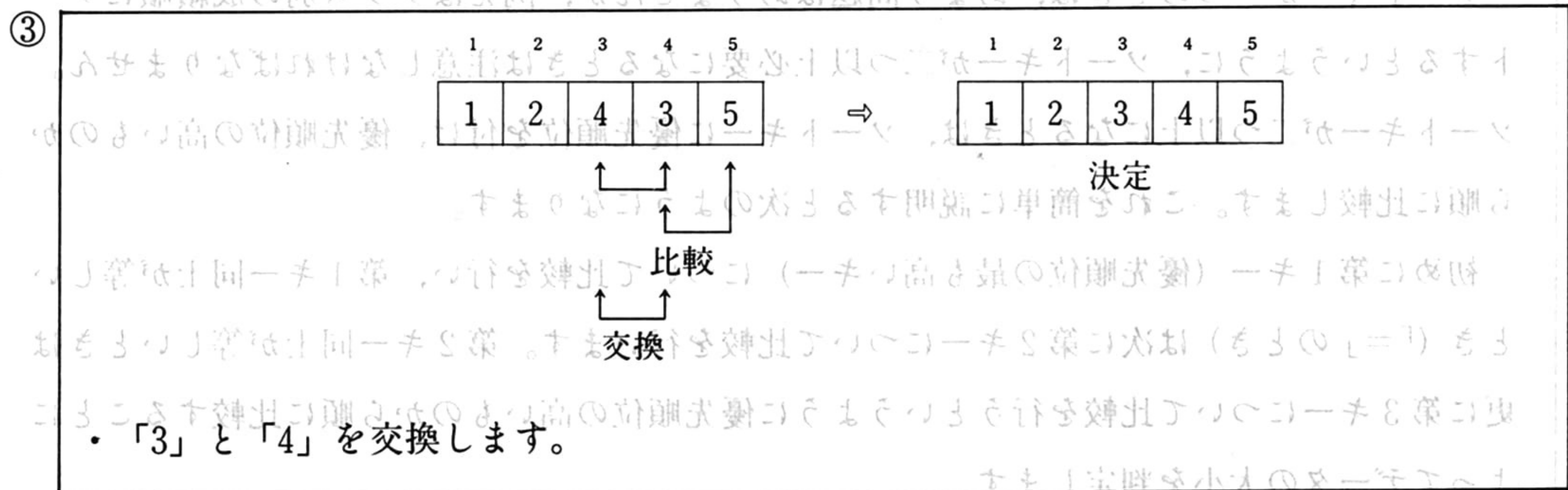
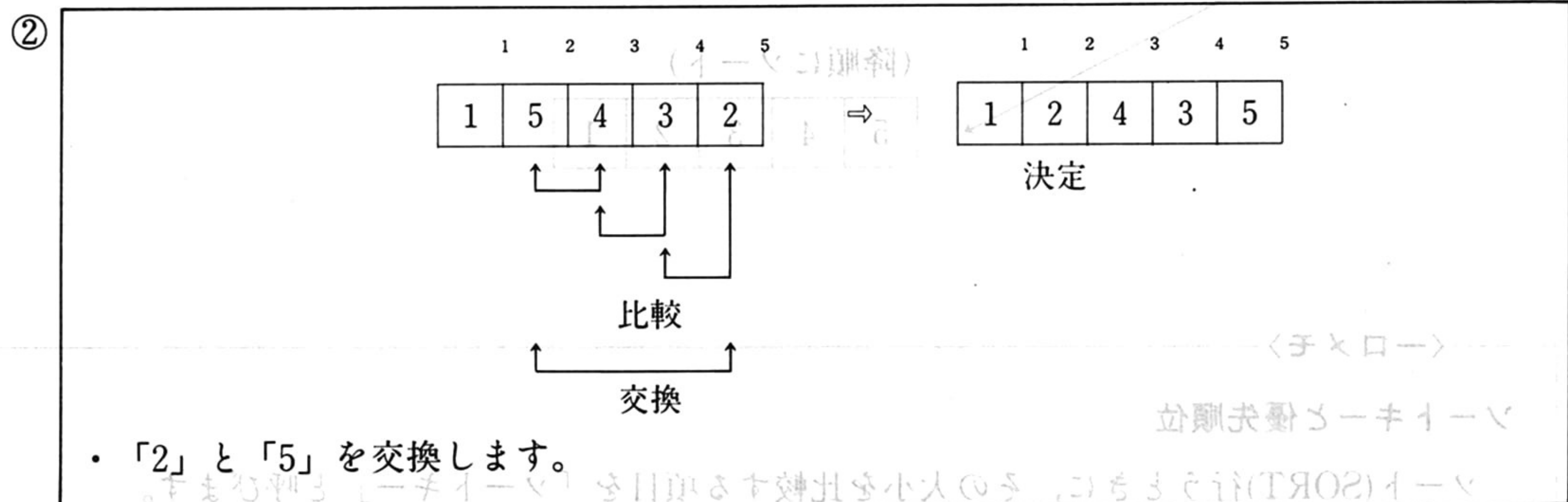
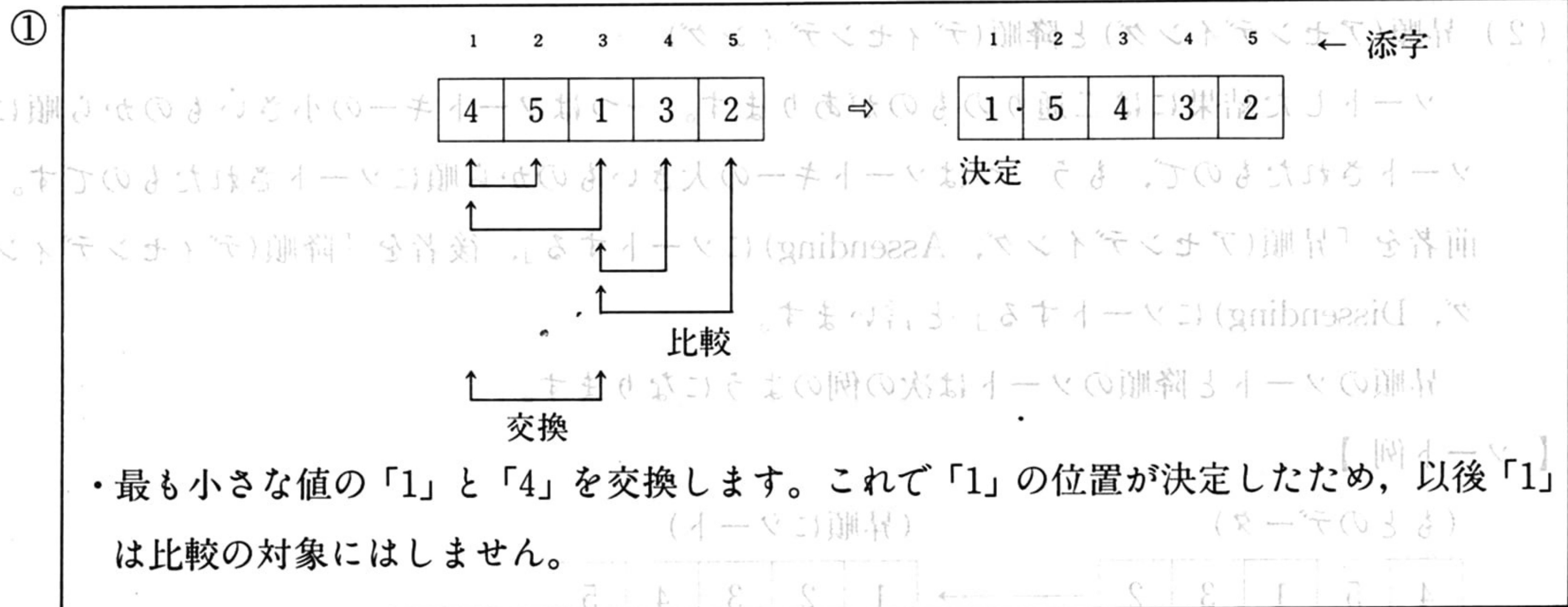
1	2	3	4	5
1	2	3	4	5

決定

比較

交換

・「3」と「4」を交換します。

⇩

④

1	2	3	4	5
1	2	3	4	5

比較

⇨

1	2	3	4	5
1	2	3	4	5

決定　決定

・データの交換は行われません。

（2） 隣接交換法(一方向型)

隣接交換法は，隣り合う二つのデータを比較・交換してソートして行く方法で，「一方向型」と「往復型」があります。一方向型とは，比較・交換の対象となる位置の移動が，添字の小から大または大から小へ一方向でしか行われないものをいいます。往復型とは，比較・交換の対象となる位置の移動が往復に行われるものです。

隣接交換法一方向型で昇順にソートする例を次に示します。

【 隣接交換法一方向型の例 】

①

1	2	3	4	5
4	5	1	3	2

比較・交換

⟶

(位置の移動)

⇨

1	2	3	4	5
4	1	3	2	5

決定

・隣り合う二つのデータを比較し，左のデータより右のデータが小さければデータを交換します。

・比較・交換の対象となる位置を一つずつ右へ移動せさて，データがなくなるまで繰り返します。

・一回の移動(パス)では，最大の値「5」の位置が決定します。

⇩

②

1	2	3	4	5
4	1	3	2	5

比較・交換

⟶

⇨

1	2	3	4	5
1	3	2	4	5

決定

・「4」の位置が決定します。

⇩

③

1	2	3	4	5
1	3	2	4	5

⇒

1	2	3	4	5
1	2	3	4	5

決定

比較・交換

⟶

・「3」の位置が決定します。

⇩

④

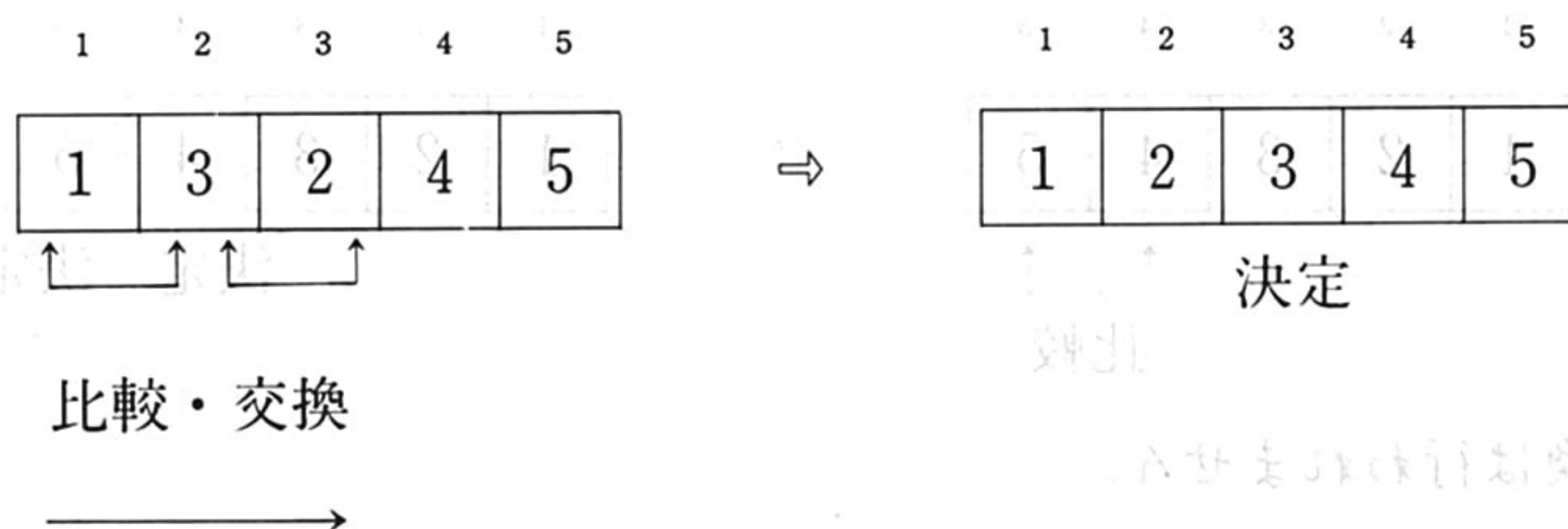
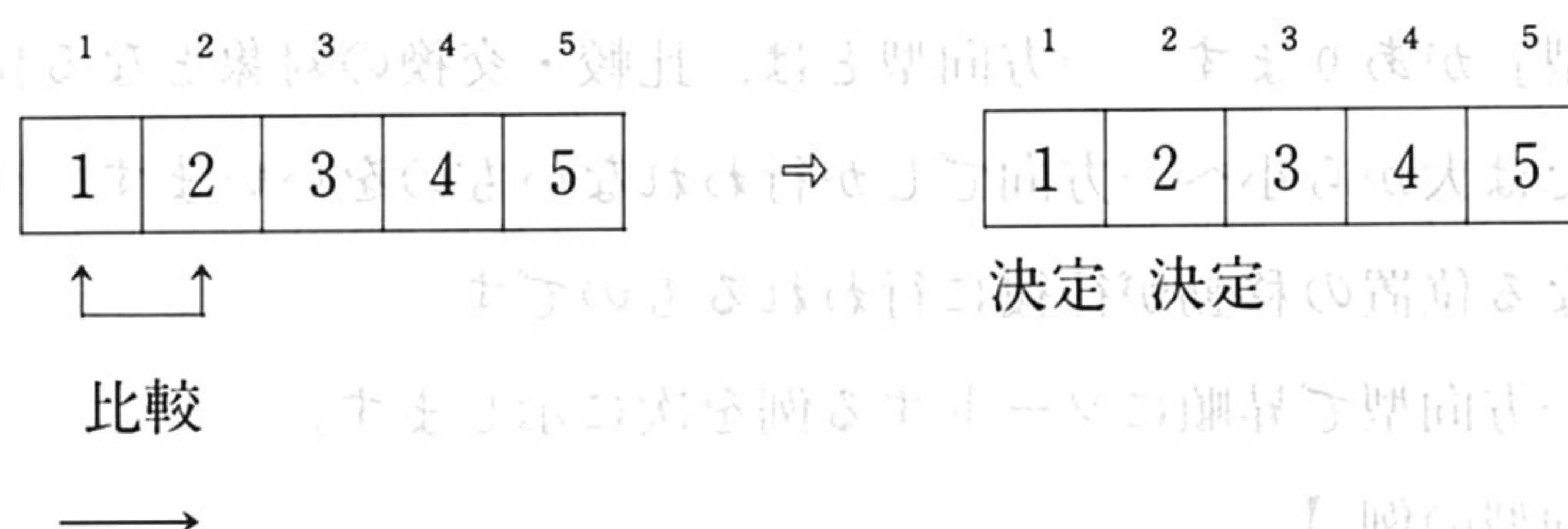
1	2	3	4	5
1	2	3	4	5

⇒

1	2	3	4	5
1	2	3	4	5

決定 決定

比較

⟶

・「1」と「2」の位置が決定します。

・データの交換はありません。

（3） 隣接交換法(往復型)

隣接交換法往復型の昇順にソートする例を次に示します。

【 隣接交換法往復型の例 】

①

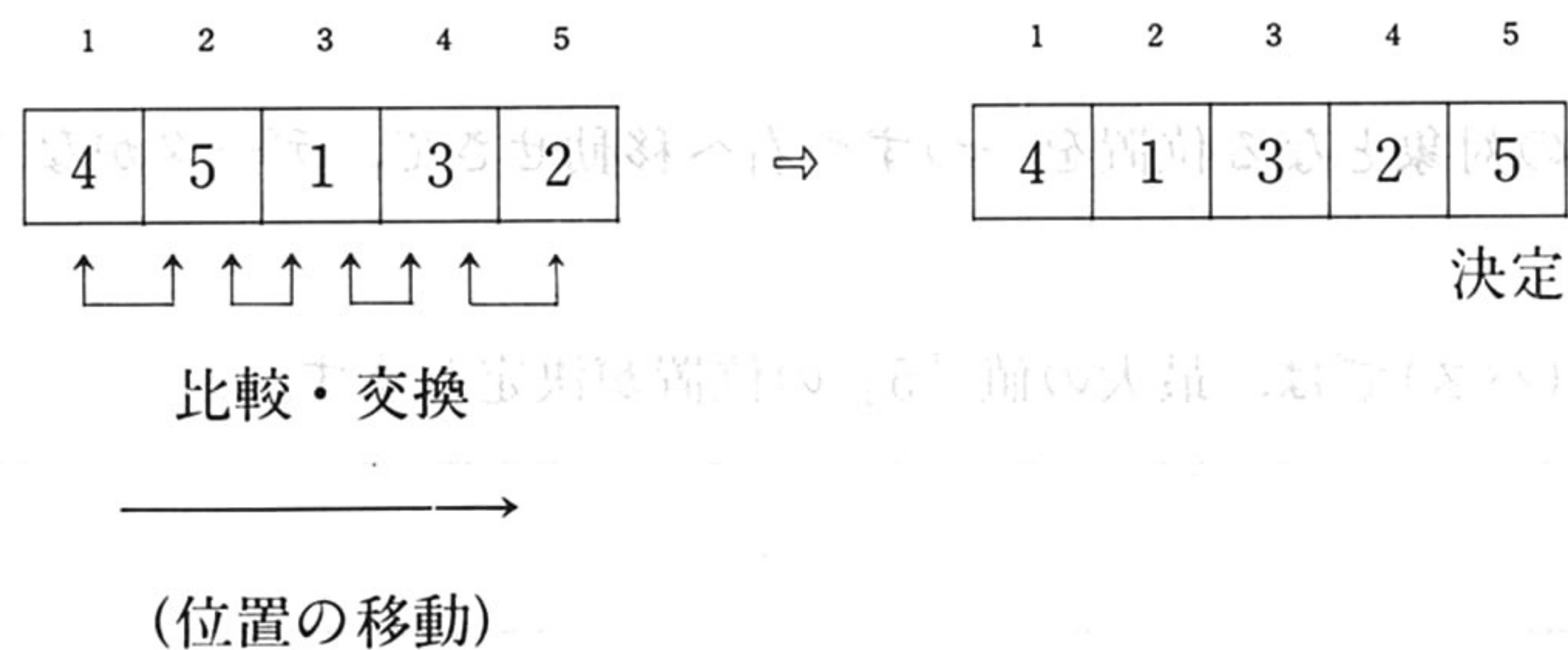
1	2	3	4	5
4	5	1	3	2

⇒

1	2	3	4	5
4	1	3	2	5

決定

比較・交換

⟶

(位置の移動)

・隣り合う二つのデータを比較し，左のデータより右のデータが小さければデータを交換します。

・一回目のパスでは，比較・交換の対象となる位置を左から右へ移動させています。(往)

・「5」の位置が決定します。

⇩

②

1	2	3	4	5
4	1	3	2	5

⇨

1	2	3	4	5
1	4	2	3	5

決定

比較・交換

←

(位置の移動)

二回目のパスでは，比較・交換の対象となる位置を右から左へ移動させています。(復)

「1」の位置が決定します。

⇩

③

1	2	3	4	5
1	4	2	3	5

⇨

1	2	3	4	5
1	2	3	4	5

決定

比較・交換

→

(位置の移動)

・三回目のパスでは，左から右へ移動しています。(往)

・「4」の位置が決定します。

⇩

④

1	2	3	4	5
1	2	3	4	5

⇨

1	2	3	4	5
1	2	3	4	5

決定 決定

比較

←

(位置の移動)

・「2」と「3」の位置が決定します。

（4） 選択法

ここでは，選択法のうちのリニア・セレクション・ソート(Linear Selection Sort)について説明します。

選択法は二つの配列を使い，昇順のソートの場合，ソートキーの一番小さなものを配列Aから取り出して配列Bに入れ，配列Aの取り出された場所には最大値を入れておきます。こ

のようにして，すべてのデータが取り出されるまで繰り返します。

選択法のリニア・セレクション・ソートの昇順のときの例を次に示します。

【 選択法の例 】

①

	1	2	3	4	5
A	4	5	1	3	2
B					

←「99」 ⇨

	1	2	3	4	5
A	4	5	99	3	2
B	1				

決定

・配列Aの最小値「1」を配列Bの先頭に移します。

・配列Aの「1」が記憶されている場所に最大値(ここでは「99」)を入れます。

⇩

②

	1	2	3	4	5
A	4	5	99	3	2
B	1				

←「99」 ⇨

	1	2	3	4	5
A	4	5	99	3	99
B	1	2			

決定

・配列Aの最小値「2」を配列Bに移します。

・配列Aの「2」が記憶されている場所に「99」を入れます。

⇩

③

	1	2	3	4	5
A	4	5	99	3	99
B	1	2			

←「99」 ⇨

	1	2	3	4	5
A	4	5	99	99	99
B	1	2	3		

決定

・配列Aの最小値「3」を配列Bに移します。

・配列Aの「3」が記憶されている場所に「99」を入れます。

⇩

④

	1	2	3	4	5
A	4	5	99	99	99
B	1	2	3		

「99」

⇨

	1	2	3	4	5
A	99	5	99	99	99
B	1	2	3	4	

決定

・配列Aの最小値「4」を配列Bに移します。

・配列Aの「4」が記憶されている場所に「99」を入れます。

⇩

⑤

	1	2	3	4	5
A	99	5	99	99	99
B	1	2	3	4	

「99」

⇨

	1	2	3	4	5
A	99	99	99	99	99
B	1	2	3	4	5

決定

・配列Aの最小値「5」を配列Bに移します。

・配列Aの「5」が記憶されている場所に「99」を入れます。

・配列Aの内容がすべて最大値「99」になったときにソートが終了します。

（5） 挿入法

挿入法は，配列Aのソートキーの値が配列Bの値より小さくなる場所に配列Aのデータを挿入していく方法です。

挿入法の例を次に示します。

【 挿入法の例 】

①

	1	2	3	4	5
A	4	5	1	3	2
B					

↓

⇨

	1	2	3	4	5
A	4	5	1	3	2
B	4				

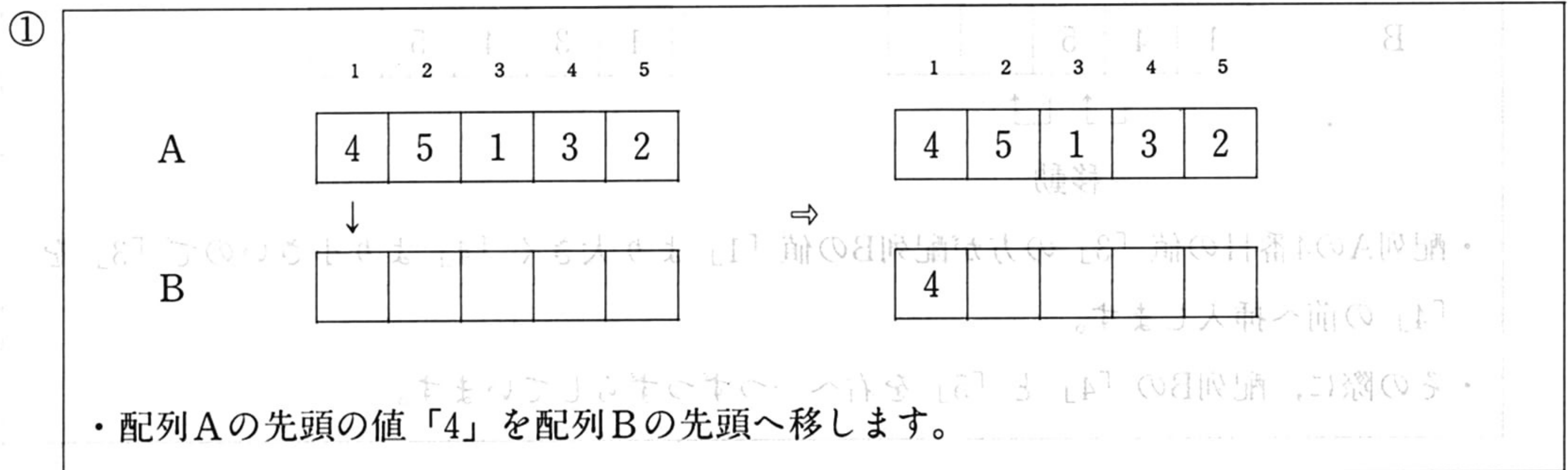
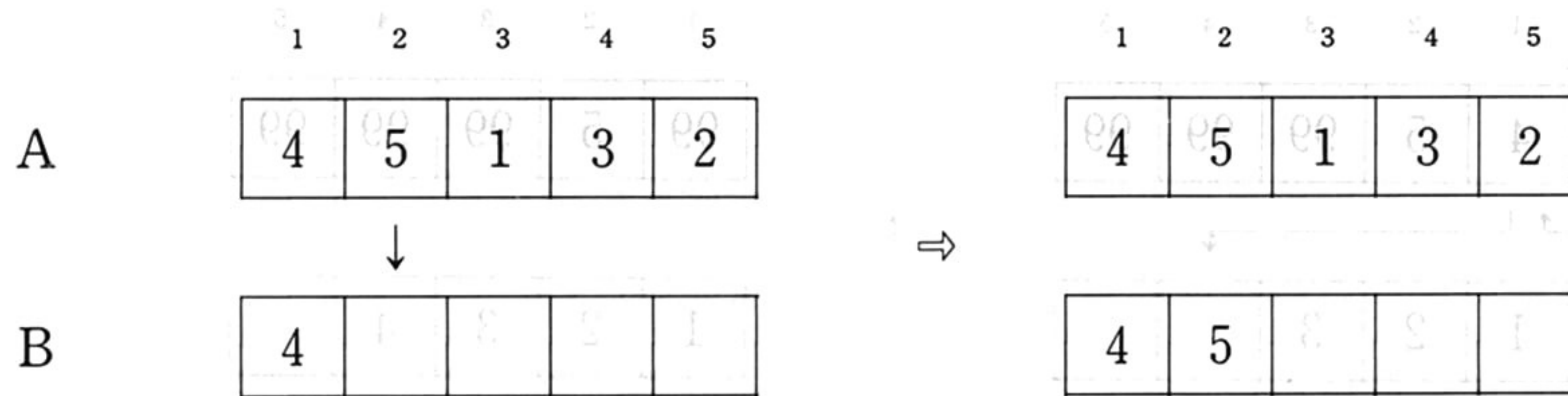
・配列Aの先頭の値「4」を配列Bの先頭へ移します。

⇩

②

	1	2	3	4	5
A	4	5	1	3	2
B	4				

⇨

	1	2	3	4	5
A	4	5	1	3	2
B	4	5			

・配列Aの2番目の値「5」より配列Bの値「4」の方が小さいので「5」を「4」の後に移します。

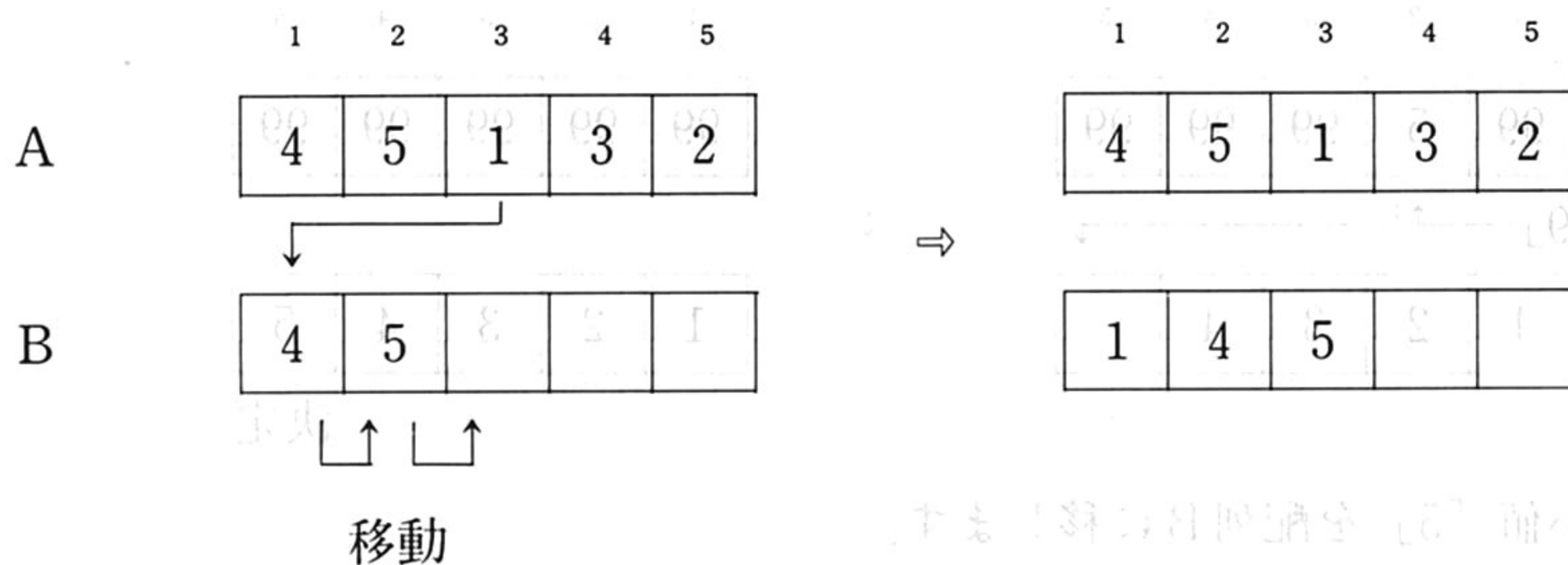
⇩

③

	1	2	3	4	5
A	4	5	1	3	2
B	4	5			

移動

⇨

	1	2	3	4	5
A	4	5	1	3	2
B	1	4	5		

・配列Aの3番目の値「1」の方が配列Bの値「4」よりも小さいので「1」を「4」の前へ挿入します。

・その際に，配列Bの「4」と「5」を右へ一つずつずらしています。

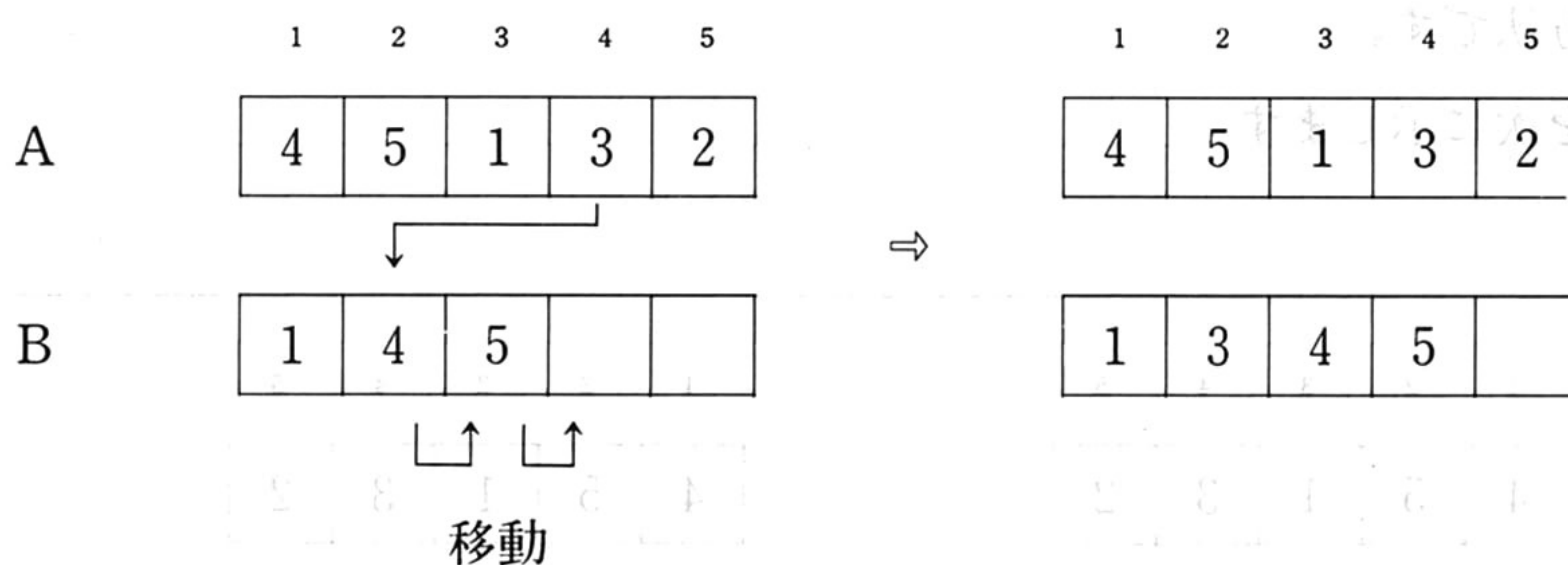
⇩

④

	1	2	3	4	5
A	4	5	1	3	2
B	1	4	5		

移動

⇨

	1	2	3	4	5
A	4	5	1	3	2
B	1	3	4	5	

・配列Aの4番目の値「3」の方が配列Bの値「1」より大きく「4」より小さいので「3」を「4」の前へ挿入します。

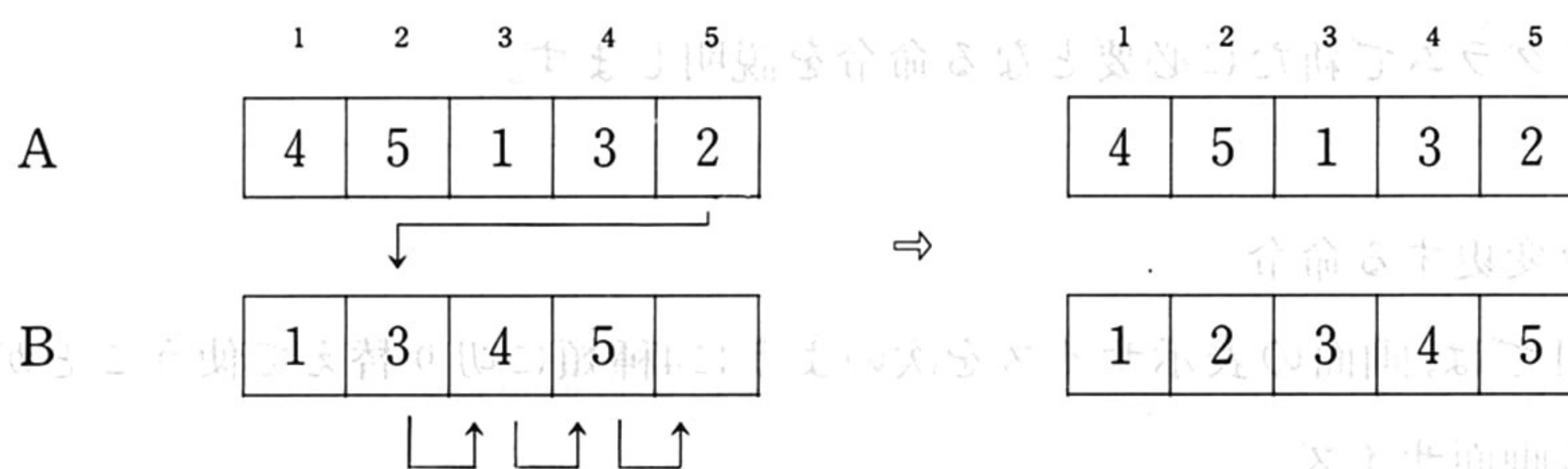
・その際に，配列Bの「4」と「5」を右へ一つずつずらしています。

⇩

⑤

	1	2	3	4	5
A	4	5	1	3	2
B	1	3	4	5	

⇨

	1	2	3	4	5
A	4	5	1	3	2
B	1	2	3	4	5

移動

・配列Aの最後の値「2」の方が配列Bの値「1」より大きく「3」より小さいので「2」を「3」の前へ挿入します。

・その際に，配列Bの「3」，「4」，「5」を右へ一つずつずらしています。

・ソートが終了しても，配列Aの内容はソート前と変わりません。

ソートの方法を5種類説明しましたが，ソートにはまだほかにもさまざまな方法があります。そこで，このテキストでは最初に説明した「逐次決定法」を使うことにします。なお，その他の方法について自分で調べてみるのも，またよい勉強になることでしょう。

4.1.5 プログラムに必要な命令の説明

この成績処理プログラムで新たに必要となる命令を説明します。

(1) 画面サイズを変更する命令

PC-8801mkIIでは,画面の表示サイズを次のように4種類に切り替えて使うことができます。

・使用できる画面サイズ

横40文字×縦20行

40 × 25

80 × 20

80 × 25

画面サイズを変更するには「WIDTH」コマンドを使います。WIDTHコマンドの一般形式と書き方例は次のとおりです。

(WIDTHコマンドの説明)

<一般形式>

WIDTH 桁数,行数

・テキスト画面に表示できる文字のサイズを設定します。

・桁数には1行に表示できる文字数を指定します。指定できる値は「40」と「80」です。

・行数には画面に表示できる行数を指定します。指定できる値は「20」と「25」です。

・行数を省略して指定すると,そのときの画面の行数のまま変化しません。

・このコマンドを実行すると,画面クリアとスクロールウインドウの初期値セットも同時に行われます。

・画面サイズの初期値は40×20です。

<書き方例>

① WIDTH 40,20　　画面サイズを40×20にします。

② WIDTH 80,25　　画面サイズを80×25にします。

③ WIDTH 80　　現在の画面の行数が20のときは画面サイズを80×20に,行数が25のときは80×25にします。

(2) ファンクションキーの内容を表示したり消したりする場合

ファンクションキーの内容を画面の最下行に表示したり消したりするには,「CONSOLE」命令を使います。CONSOLE命令の一般形式と書き方例は次のとおりです。

(CONSOLE命令の説明)

<一般形式>

CONSOLE スクロール開始行, スクロール行数, ファンクションキー表示スイッチ,
カラー/白黒モード

・テキスト画面のモードや条件の設定を行います。

・スクロール開始行とスクロール行数を指定することで, スクロールする画面の範囲を設定します。

・スクロール開始行で指定できる範囲は0~24です。

・スクロール行数で指定できる範囲は1~25です。

・スクロール開始行とスクロール行数の指定は同時に省略することができます。このときは, スクロールの状態が前のままになります。

・ファンクションキーの表示スイッチに「0」を指定すると, ファンクションキーの内容の表示が消えます。

・ファンクションキー表示スイッチに「1」を指定すると, ファンクションキーの内容を表示します。

・ファンクションキー表示スイッチの指定を省略すると, ファンクションキーの内容の表示の状態は変わりません。

・カラー/白黒スイッチに「0」を指定すると, 画面は白黒モードになります。

・カラー/白黒スイッチに「1」を指定すると, 画面はカラーモードになります。

・カラー/白黒スイッチの指定を省略すると, 画面のカラー/白黒のモードは変わりません。

・WIDTHコマンドを実行すると, この命令で指定していたスクロール領域は解除され, 0行目から最後の行まですべてスクロールするようになります。

・初期値は CONSOLE 0,25,1,0 です。

<書き方例>

・ CONSOLE ,,0 画面に表示しているファンクションキーの内容を消します。

(3) 変数の内容を交換する命令

ソートを行おうとすると, 配列要素どうしのデータの交換が必要になります。二つの変数の内容を交換するときは, もう一つの作業用の変数を使います。

【 データを交換する方法 】

①

A 5　B 10　⇨　A 5　B 10

W (空)　　　　W 5

・変数Aの内容を作業用変数Wへ移します。

⇩

②

A 5 ← B 10　⇨　A 10　B 10

W 5　　　　W 5

・次に変数Bの内容を変数Aへ移します。

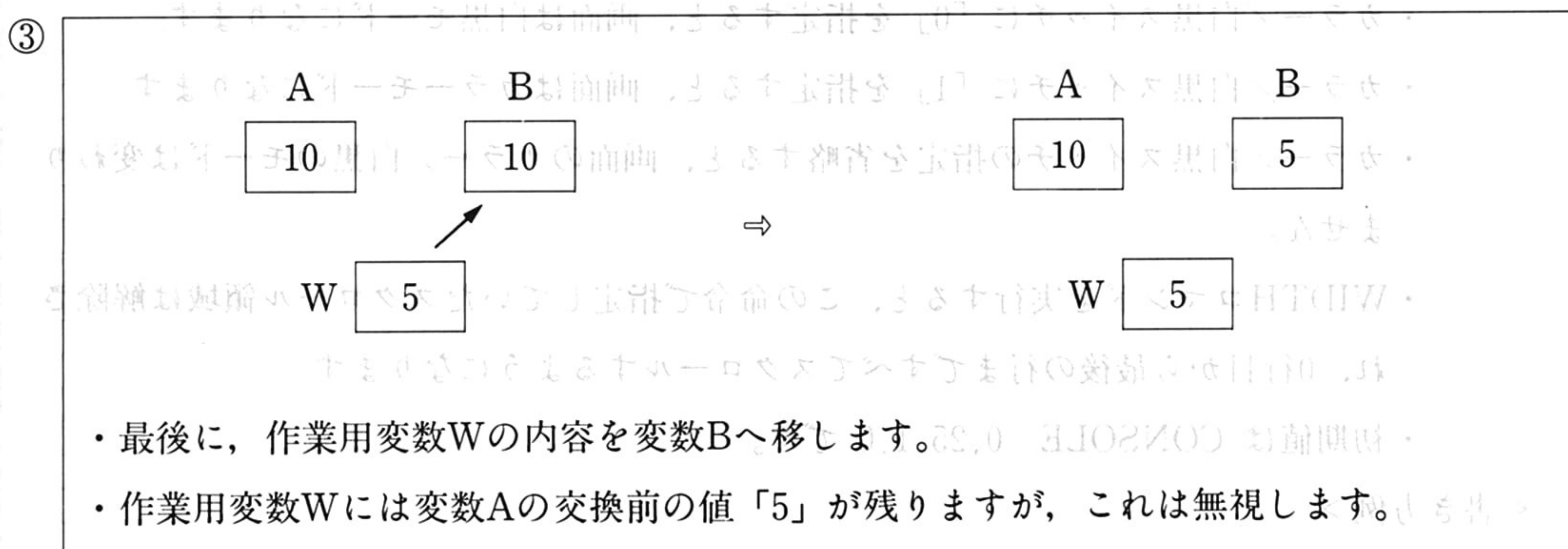
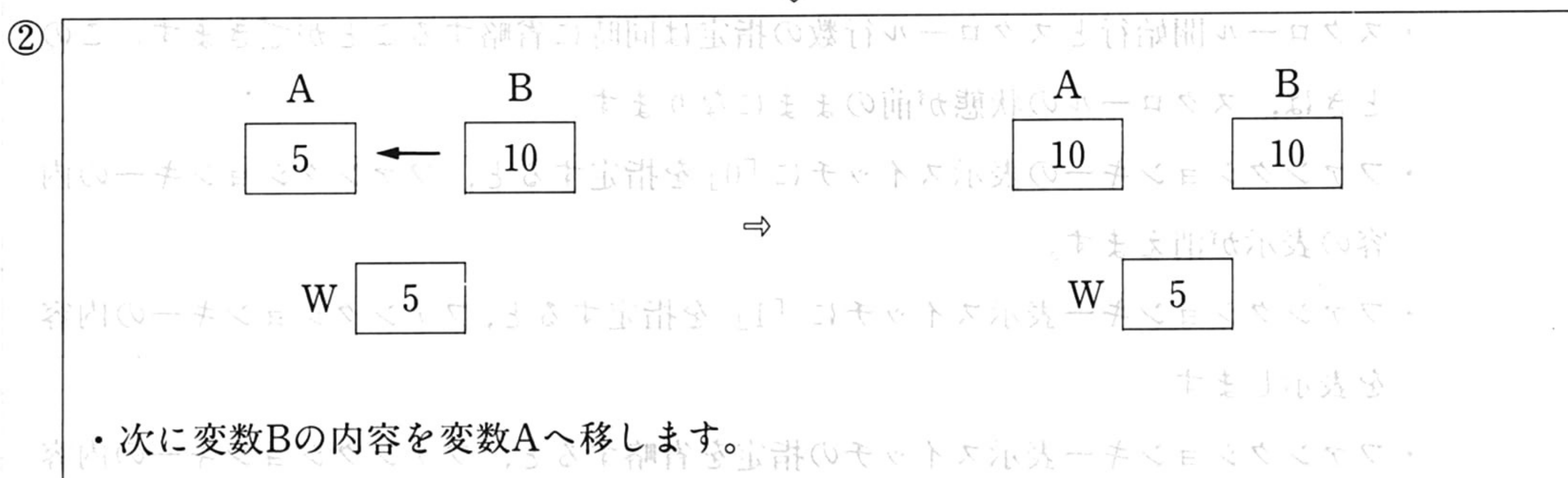
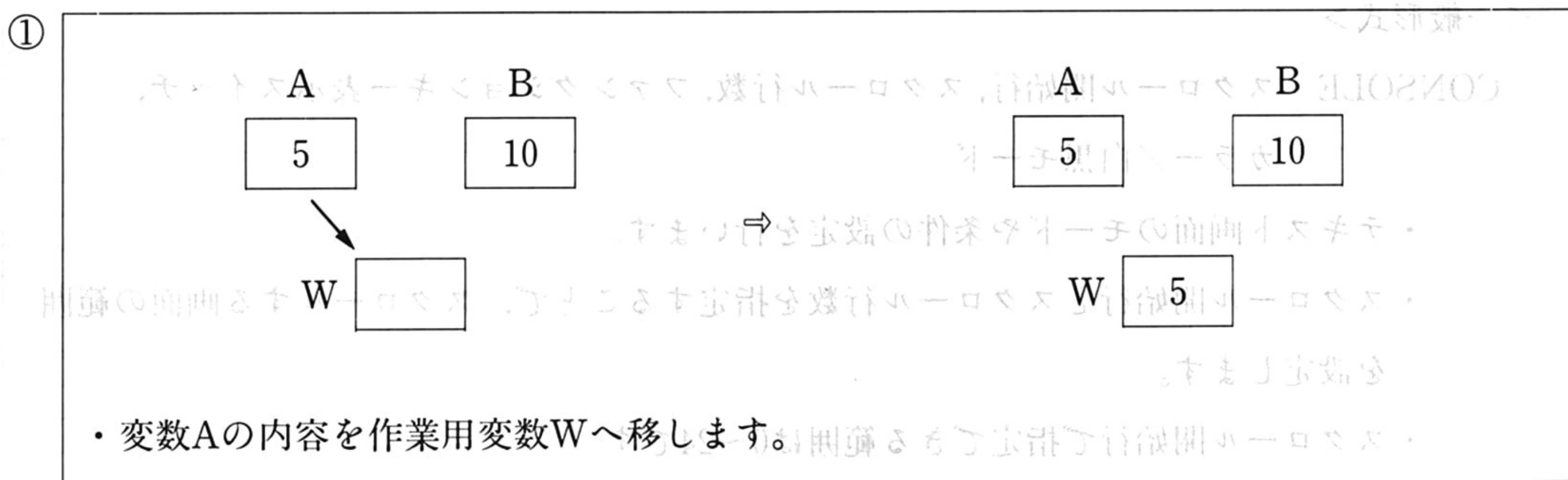
⇩

③

A 10　B 10　⇨　A 10　B 5

W 5　　　　W 5

・最後に，作業用変数Wの内容を変数Bへ移します。

・作業用変数Wには変数Aの交換前の値「5」が残りますが，これは無視します。

ソートの変数の内容を交換するたびにいちいち上の動作を意識してプログラムを作成していたのでは，少々面倒くさくなります。 そこで，PC-8801mkIIには上の動作を簡単に行う便利な命令があります。

二つの変数の内容を交換するには「SWAP」命令を使います。SWAP命令の一般形式と書き方例は次のとおりです。

（SWAP命令の説明）

＜一般形式＞

SWAP　変数名1,変数名2

・二つの変数の内容を交換します。

＜書き方例＞

① SWAP　A,B　　　数値変数Aと数値変数Bの内容を交換します。

② SWAP　X＄,Y＄　　　文字変数X＄と文字変数Y＄の内容を交換します。

（4）　表示や印字の位置を決める関数

データを画面やプリンタへ出力するときに自由に好きな位置へ出力できたら，表示や印字の編集が容易になります。PC-8801mkIIには，画面やプリンタへの出力の際に，データを出力する位置を決定する関数があります。

データの表示や印字のときに位置設定をするには「TAB」関数を使います。TAB関数の一般形式と書き方例は次のとおりです。

（TAB関数の説明）

＜一般形式＞

TAB(数式)

・一行中の数式で指定された位置まで空白(スペース)を空けます。

・位置の指定は1行中の左端を「0」として数えた値を指定します。

・TAB関数は，PRINT命令またはLPRINT命令の中で使用します。

・数式で指定できる値の範囲は－32768～32767ですが，普通，画面では0～39または0～79，プリンタでは0～79(拡大印字モードのときは0～39,縮小印字モードのときは0～135)の範囲で指定します。

＜書き方例＞

① PRINT　TAB(20)；"ABC"

画面の21桁目から「ABC」という文字列を表示します。

② LPRINT　TAB(5)；"ABC"；TAB(30)；"XYZ"

プリンタの6桁目から「ABC」という文字列を，31桁目から「XYZ」という文字列を印字します。

③ LPRINT　TAB(A)；X

プリンタに変数Aの内容で示される位置から変数Xの内容を印字します。

（5） IF命令で条件式を複数個使うには

今までは一つのIF命令の中では一つの条件式しか使っていませんでしたが，IF命令の中では，複数個の条件式が指定できます。IF命令の中で条件式を複数個指定するには，「論理演算子」を使います。論理演算子には次のものがあります。

・NOT（否定）

・AND（論理積）

・OR（論理和）

・XOR（排他的論理和）

・IMP（包含）

・EQV（同値）

ここでは「AND」と「OR」について説明します。他のものはほとんど使うことがないので，ここでの説明は省きます。

① 「AND」の考え方

ある仕事を行うときに，二つ以上の条件を判断して処理を行うことがあります。

たとえば，「性別が男でしかも年令が20才以上の人」という条件文をANDを使って表現すると，次のようになります。

性格＝男性　AND　年令＞＝20

ANDを使ったIF命令の一般形式と，書き方列およびそのときのフローチャート列は次のとおりです。

（ANDを使ったIF文の説明）

＜一般形式＞

IF　条件式　AND　条件式　AND　……　THEN　命令文1　ELSE　命令文2

・条件式がすべて「真」のときに命令文1を，「偽」のときに命令文2を実行します。

＜書き方列＞

・　IF　A＝0　AND　B＝0　THEN　A＝1：B＝1　ELSE　A＝0：B＝0

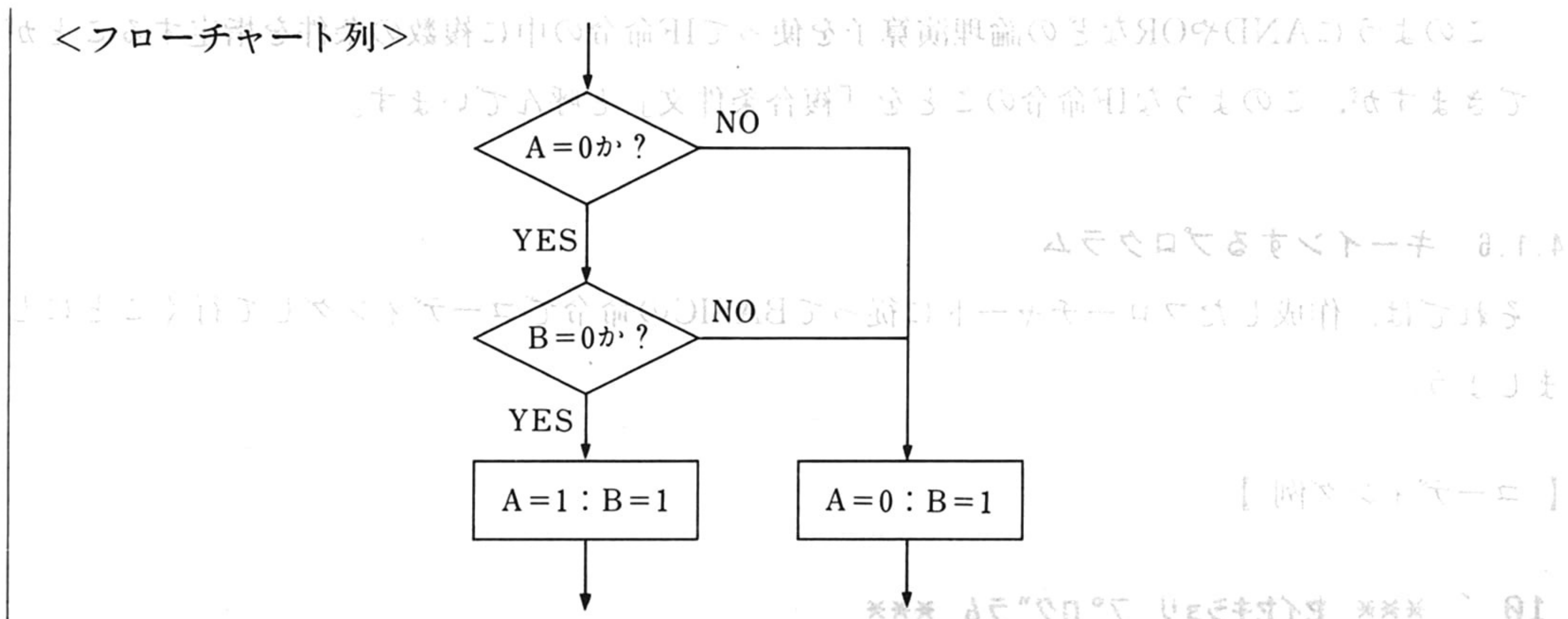
AもBも「0」ならばそれぞれに「1」を入れます。AかBのいずれか一方または両方が「0」でなければそれぞれに「0」を入れます。

＜フローチャート列＞

A＝0か？ (YES → B＝0か？ / NO → A＝0：B＝1)

B＝0か？ (YES → A＝1：B＝1 / NO → A＝0：B＝1)

A＝1：B＝1

A＝0：B＝1

② 「OR」の考え方

また，ANDに対して「OR」は，複数の条件のうち，いずれか一つでも条件を満足すると処理を行うときに使います。

たとえば，「赤か白かの花」という条件文をORを使って表現すると，次のようになります。

花の色＝赤　OR　花の色＝白

ORを使ったIF命令の一般形式と，書き方例およびそのときのフローチャート例は次のとおりです。

(ORを使ったIF文の説明)

＜一般形式＞

IF　条件式　OR　条件式　OR　……　THEN　命令文1　ELSE　命令文2

・条件式のいずれかひとつが「真」ならば命令文1を，すべて「偽」ならば命令文2を実行します。

＜書き方例＞

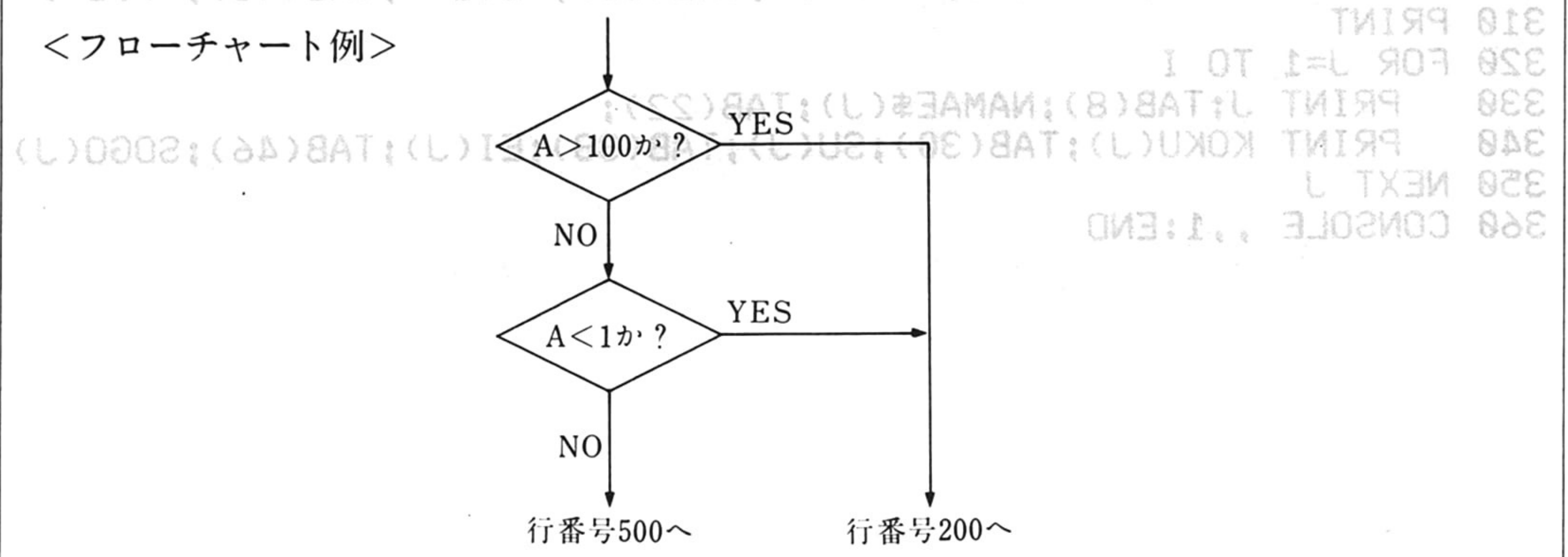
・　IF　A＜100　OR　THEN　500　ELSE　200

Aが「100」より大きいかまたは「1」より小さければ行番号500へ分岐します。Aが1と100の間であれば行番号200へ分岐します。

＜フローチャート例＞

A＞100か？ (YES → 行番号200へ / NO → A＜1か？)

A＜1か？ (YES → 行番号200へ / NO → 行番号500へ)

行番号500へ　　行番号200へ

このようにANDやORなどの論理演算子を使ってIF命令の中に複数の条件を指定することができますが，このようなIF命令のことを「複合条件文」と呼んでいます。

4.1.6 キーインするプログラム

それでは，作成したフローチャートに従ってBASICの命令でコーディングして行くことにしましょう。

【 コーディング例 】

```
10 ' *** ｾｲｾｷｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ***
20 N=10:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
30 CONSOLE ,,0
40 ' ** ﾆｭｳﾘｮｸ ﾙｰﾁﾝ **
50 WIDTH 40,20
60 FOR I=1 TO N
70   INPUT "ﾅﾏｴ (END=[E]) ";NAMAE$(I)
80   IF NAMAE$(I)="E" OR NAMAE$(I)="e" THEN *SORT.RTN
90   INPUT "ｺｸｺﾞ  = ",KOKU(I)
100  INPUT "ｽｳｶﾞｸ = ",SU(I)
110  INPUT "ｴｲｺﾞ  = ",EI(I):PRINT
120  SOGO(I)=KOKU(I)+SU(I)+EI(I)
130 NEXT I
140 ' ** ｿｰﾄ ﾙｰﾁﾝ **
150 *SORT.RTN:I=I-1
160 FOR J1=1 TO I-1
170   MEMO=J1
180   FOR J2=J1+1 TO I
190     IF SOGO(MEMO)<SOGO(J2) THEN MEMO=J2
200   NEXT J2
210  SWAP NAMAE$(J1),NAMAE$(MEMO)
220  SWAP KOKU(J1),KOKU(MEMO)
230  SWAP SU(J1),SU(MEMO)
240  SWAP EI(J1),EI(MEMO)
250  SWAP SOGO(J1),SOGO(MEMO)
260 NEXT J1
270 ' ** ｼｭﾂﾘｮｸ ﾙｰﾁﾝ **
280 WIDTH 80,20
290 PRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 PRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 PRINT
320 FOR J=1 TO I
330   PRINT J;TAB(8);NAMAE$(J);TAB(22);
340   PRINT KOKU(J);TAB(30);SU(J);TAB(38);EI(J);TAB(46);SOGO(J)
350 NEXT J
360 CONSOLE ,,1:END
```

【 コーディング例の説明 】

・行番号80は，行番号70のINPUT命令でエンドデータとして「E」が入力されても，「e」が入力されても入力処理が終了するように，「OR」を使った複合条件文にしています。

・行番号90,100,110のINPUT命令は，入力を促すためのメッセージ(これをプロンプト文という)の後に「？」が表示されないようにするため，プロンプト文と変数を「,」で区切っています。

・PRINT命令やLPRINT命令では，命令文を「；(セミコロン)」で終わると命令実行後の改行を行いません。そのため行番号280と行番号320は，PRINT命令を実行した後も改行しないで行番号290および行番号330のPRINT命令を実行します。PRINT命令やLPRINT命令の場合，命令文が長くなり過ぎるとこのように何行かに分けることがあります。

・フローチャートのカウンタAには変数Iを使っています。

・フローチャートのカウンタBには変数J1を使っています。

・フローチャートのカウンタCには変数J2を使っています。

・フローチャートのカウンタDには変数Jを使っています。

・ワーク変数1には変数MEMOを使っています。

・このプログラムには，一行が40文字を超えているような長い命令文もあるため，プログラムのキーインは画面を横80桁のサイズにしてから行います。

設問 26　画面サイズを横80桁にして下さい。

【 実行結果例 】

```
width 80
```

すると，画面クリアされてカーソルが小さくなります。

```
▌
```

設問 27　メモリーをクリアした後，コーディング例のプログラムをキーインして下さい。

【 実行手順 】

① メモリーをクリアするには「NEW」コマンドを使います。(2章の2.1.1 を参照)

② 行番号のキーインは「AUTO」コマンドを使うと便利です。(3章の3.4.1 を参照)

設問 28　正しくキーインされているかどうか，LISTコマンドを使ってプログラムリストを画面に表示して確認して下さい。

【 実行結果例 】

```
list
10 ' *** ｾｲｾｷｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ***
20 N=10:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
30 CONSOLE ,,0
40 ' ** ﾆｭｳﾘｮｸ ﾙｰﾁﾝ **
50 WIDTH 40,20
60 FOR I=1 TO N
70   INPUT "ﾅﾏｴ (END=[E]) ";NAMAE$(I)
80   IF NAMAE$(I)="E" OR NAMAE$(I)="e" THEN *SORT.RTN
90   INPUT "ｺｸｺﾞ  = ",KOKU(I)
100  INPUT "ｽｳｶﾞｸ = ",SU(I)
110  INPUT "ｴｲｺﾞ  = ",EI(I):PRINT
120  SOGO(I)=KOKU(I)+SU(I)+EI(I)
130 NEXT I
140 ' ** ｿｰﾄ ﾙｰﾁﾝ **
150 *SORT.RTN:I=I-1
160 FOR J1=1 TO I-1
170   MEMO=J1
180   FOR J2=J1+1 TO I
190     IF SOGO(MEMO)<SOGO(J2) THEN MEMO=J2
200   NEXT J2
210  SWAP NAMAE$(J1),NAMAE$(MEMO)
220  SWAP KOKU(J1),KOKU(MEMO)
230  SWAP SU(J1),SU(MEMO)
240  SWAP EI(J1),EI(MEMO)
250  SWAP SOGO(J1),SOGO(MEMO)
260 NEXT J1
270 ' ** ｼｭﾂﾘｮｸ ﾙｰﾁﾝ **
280 WIDTH 80,20
290 PRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 PRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 PRINT
320 FOR J=1 TO I
330   PRINT J;TAB(8);NAMAE$(J);TAB(22);
340   PRINT KOKU(J);TAB(30);SU(J);TAB(38);EI(J);TAB(46);SOGO(J)
350 NEXT J
360 CONSOLE ,,1:END
Ok
```

【 実行後の説明 】

- このプログラムのリストを一度に画面に表示しようとしても，スクロールアップして先頭から数行が消えてしまい，確認することができません。
- そこで，何行かに分けて表示するか，リスト表示中に CTRL ⊕ S キーを押して一時的に止めるかして，プログラムを確認します。
- CTRL ⊕ S を押して停止させた場合，表示を再開するには文字キーなどを押します。
- プログラム全体を確認するときは，プログラムリストをプリンタに印字します。

設問 29　LLISTコマンドを使って，プログラムリストを全部プリンタに印字して下さい。

【 実行結果例 】

```
10 ' *** ｾｲｾｷｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ***
20 N=10:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
30 CONSOLE ,,0
40 ' ** ﾆｭｳﾘｮｸ ﾙｰﾁﾝ **
50 WIDTH 40,20
60 FOR I=1 TO N
70   INPUT "ﾅﾏｴ (END=[E]) ";NAMAE$(I)
80   IF NAMAE$(I)="E" OR NAMAE$(I)="e" THEN *SORT.RTN
90   INPUT "ｺｸｺﾞ  = ",KOKU(I)
100  INPUT "ｽｳｶﾞｸ = ",SU(I)
110  INPUT "ｴｲｺﾞ  = ",EI(I):PRINT
120  SOGO(I)=KOKU(I)+SU(I)+EI(I)
130 NEXT I
140 ' ** ｿｰﾄ ﾙｰﾁﾝ **
150 *SORT.RTN:I=I-1
160 FOR J1=1 TO I-1
170   MEMO=J1
180   FOR J2=J1+1 TO I
190     IF SOGO(MEMO)<SOGO(J2) THEN MEMO=J2
200   NEXT J2
210  SWAP NAMAE$(J1),NAMAE$(MEMO)
220  SWAP KOKU(J1),KOKU(MEMO)
230  SWAP SU(J1),SU(MEMO)
240  SWAP EI(J1),EI(MEMO)
250  SWAP SOGO(J1),SOGO(MEMO)
260 NEXT J1
270 ' ** ｼｭﾂﾘｮｸ ﾙｰﾁﾝ **
280 WIDTH 80,20
290 PRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 PRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 PRINT
320 FOR J=1 TO I
330   PRINT J;TAB(8);NAMAE$(J);TAB(22);
340   PRINT KOKU(J);TAB(30);SU(J);TAB(38);EI(J);TAB(46);SOGO(J)
350 NEXT J
360 CONSOLE ,,1:END
```

4.1.7 作成したプログラムの実行

設問 30　次のデータを使って，今メモリーに記憶しているプログラムを実行して下さい。

（入力するデータ）

名前	国語	数学	英語
サッポロ　イチロウ	80	60	65
トウキョウ　ヒロシ	35	5	50

(入力するデータ)

名前	国語	数学	英語
ナゴヤ　カズコ	75	72	68
オオサカ　タツヤ	8	90	44
フクオカ　ユウタ	60	60	71
カゴシマ　ミユキ	65	85	70
E			

【 実行結果例 】

(入力時/40×20)

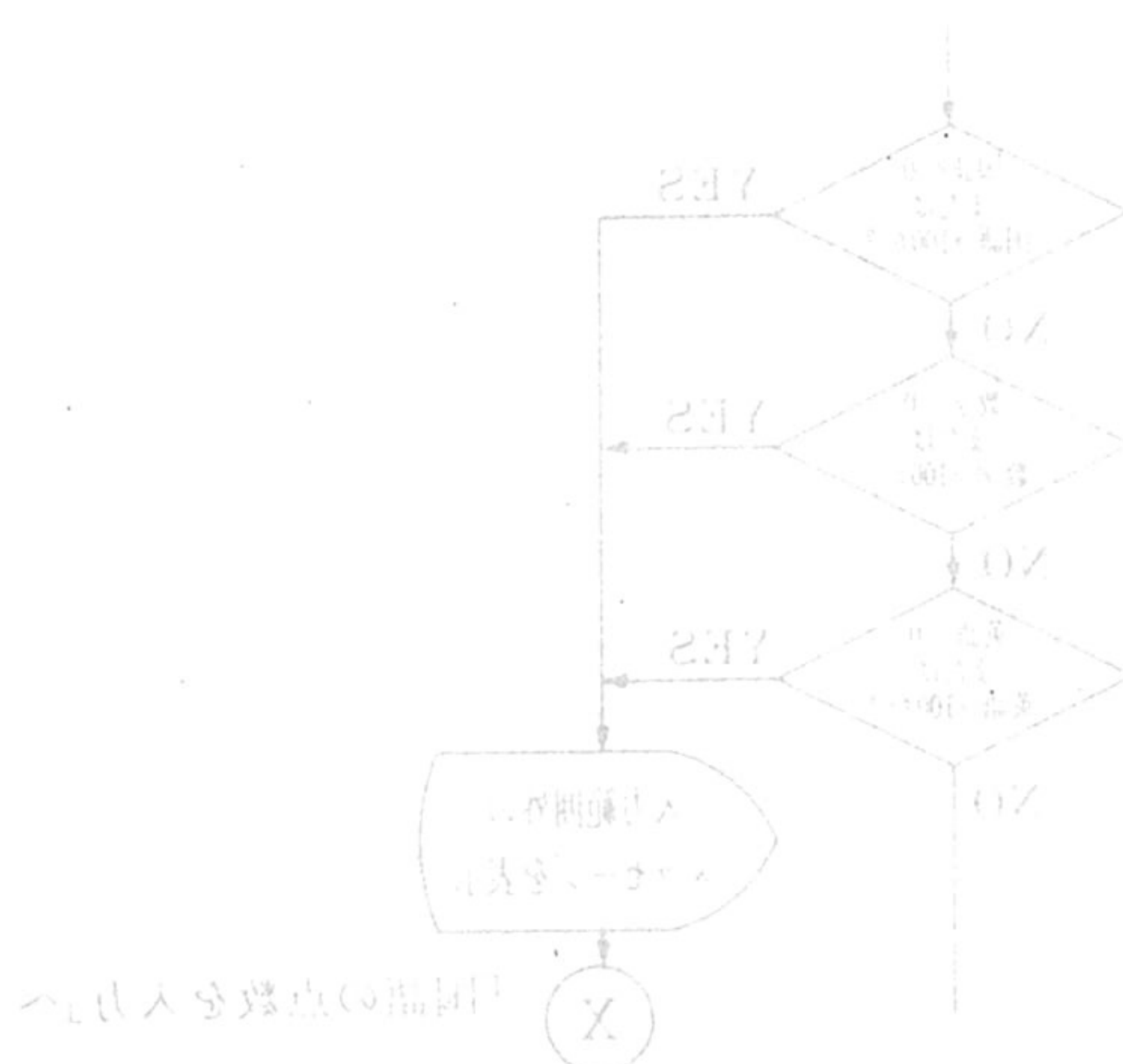
```
ﾅﾏｴ (END=[E]) ? ｻｯﾎﾟﾛ ｲﾁﾛｳ
ｺｸｺﾞ   = 80
ｽｳｶﾞｸ = 60
ｴｲｺﾞ   = 65

ﾅﾏｴ (END=[E]) ? ﾄｳｷｮｳ ﾋﾛｼ
ｺｸｺﾞ   = 35
ｽｳｶﾞｸ = 5
ｴｲｺﾞ   = 50

ﾅﾏｴ (END=[E]) ? ﾅｺﾞﾔ ｶｽﾞｺ
ｺｸｺﾞ   = 75
ｽｳｶﾞｸ = 72
ｴｲｺﾞ   = 68

ﾅﾏｴ (END=[E]) ? ｵｵｻｶ ﾀﾂﾔ
ｺｸｺﾞ   = 8
ｽｳｶﾞｸ = 90
ｴｲｺﾞ   = 44

ﾅﾏｴ (END=[E]) ? ﾌｸｵｶ ﾕｳﾀ
ｺｸｺﾞ   = 60
ｽｳｶﾞｸ = 60
ｴｲｺﾞ   = 71

ﾅﾏｴ (END=[E]) ? ｶｺﾞｼﾏ ﾐﾕｷ
ｺｸｺﾞ   = 65
ｽｳｶﾞｸ = 85
ｴｲｺﾞ   = 70

ﾅﾏｴ (END=[E]) ? E
```

（出力時/80×40）

```
ｼﾞｭﾝｲ     ﾅﾏｴ            ｺｸｺﾞ    ｽｳｶﾞｸ    ｴｲｺﾞ    ｿｳｺﾞｳ
  1       ｶｺﾞｼﾏ ﾐﾕｷ      65      85       70      220
  2       ﾅｺﾞﾔ ｶｽﾞｺ      75      72       68      215
  3       ｻｯﾎﾟﾛ ｲﾁﾛｳ     80      60       65      205
  4       ﾌｸｵｶ ﾕｳﾀ       60      60       71      191
  5       ｵｵｻｶ ﾀﾂﾔ       8       90       44      142
  6       ﾄｳｷｮｳ ﾋﾛｼ      35      5        50      90
Ok
```

設問 31　正しく実行できていたら，ドライブ2のフロッピーディスクへ「SEISEK.1」という名前でセーブして下さい。

【 実行結果例 】

```
save "2:SEISEK.1"
Ok
```

4.2　より良いプログラムにするには

今まで作成してきたプログラムでも，簡単な処理をするためには十分役に立ちます。しかし，予想した範囲外のデータ，たとえば，0点未満や100点を超える点数が入力されたときや，出力データの編集などはできません。そこでこの項では，入力データの範囲をプログラムでチェックしたり，数値データの桁揃えなどの編集の方法学習します。

4.2.1　入力データをチェックする

(1)　プログラムの修正

この成績修正プログラムは，入力する国・数・英の点数の範囲が0～100であるにもかかわらず，0点未満の点数や100点を超える点数も処理できるようになっています。

そこで，入力された点数が0～100の範囲内に入っているかどうかチェックする処理を追加します。追加する条件は次のとおりです。

【 追加する処理の条件 】

・国・数・英の点数の入力が終わったら，三つのデータがそれぞれ0～100の範囲内がどうかをチェックします。

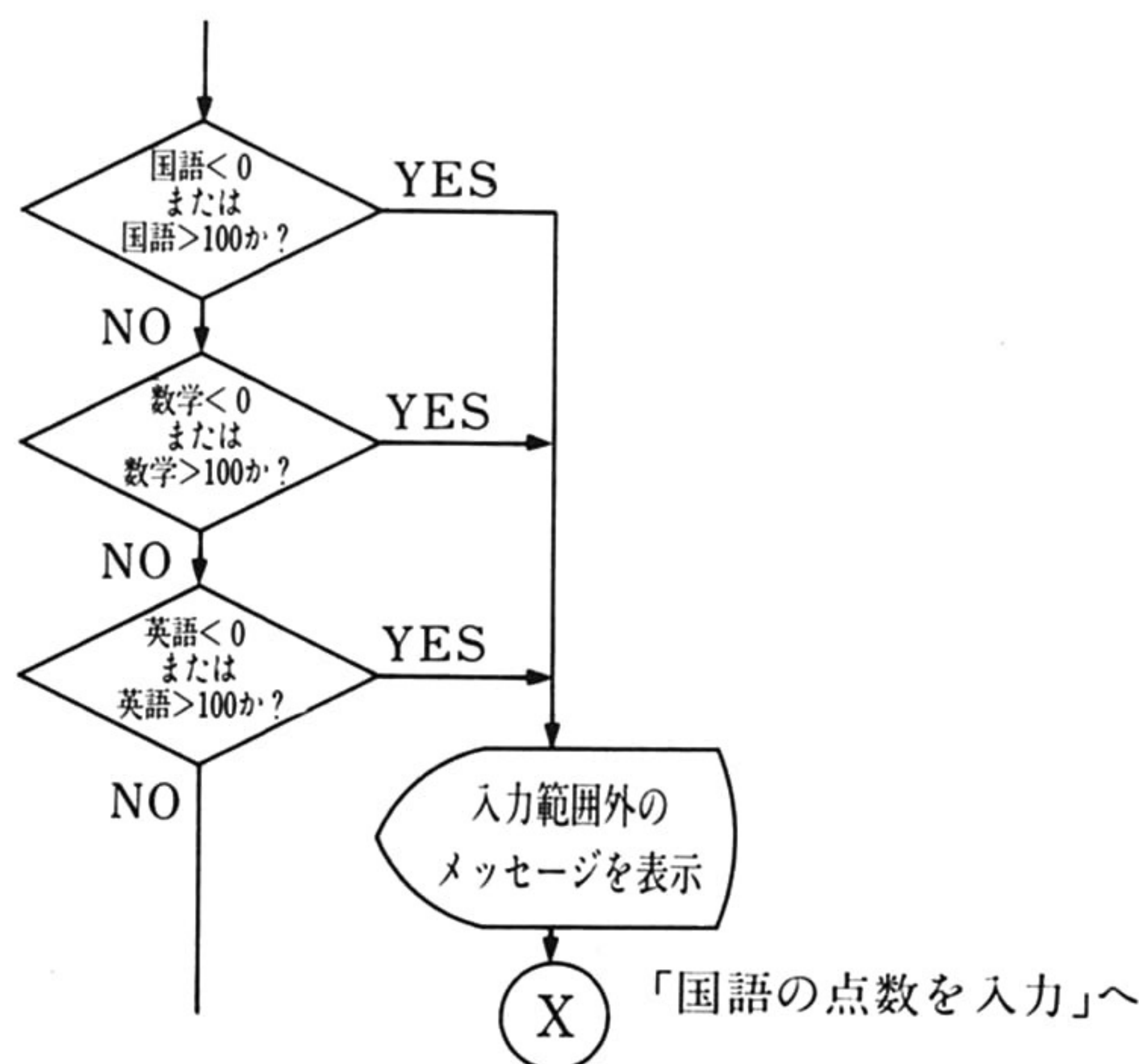
・三つのデータがすべて0～100の範囲内に入っていれば，三つのデータの合計を求めます。

・0～100の範囲外のディスクが入力されたら，入力範囲外を示すメッセージを表示します。

・三つのデータの中で一つでも0～100の範囲に入っていなければ，国語の点数の入力からやり直します。

続いて，上の条件に従ってフローチャートを修正します。

【 追加するフローチャート 】

国語＜0 または 国語＞100か？　YES
NO
数学＜0 または 数学＞100か？　YES
NO
英語＜0 または 英語＞100か？　YES
NO
入力範囲外のメッセージを表示
X　「国語の点数を入力」へ

修正後のフローチャートは次のようになります。

【 修正後のフローチャート 】

開　始

配列の宣言

ファンクションキーの内容の表示を消す

画面サイズの変更 ---- 横40文字 縦20行

カウンタAを初期値クリア

名前の入力

入力データはエンド・データか?　YES / NO

(X)

国語の点数を入力

数学の点数を入力

英語の点数を入力

国語<0 または 国語>100か?　YES / NO

数学<0 または 数学>100か?　YES / NO

英語<0 または 英語>100か?　YES / NO

入力範囲外のメッセージを表示 → (X)

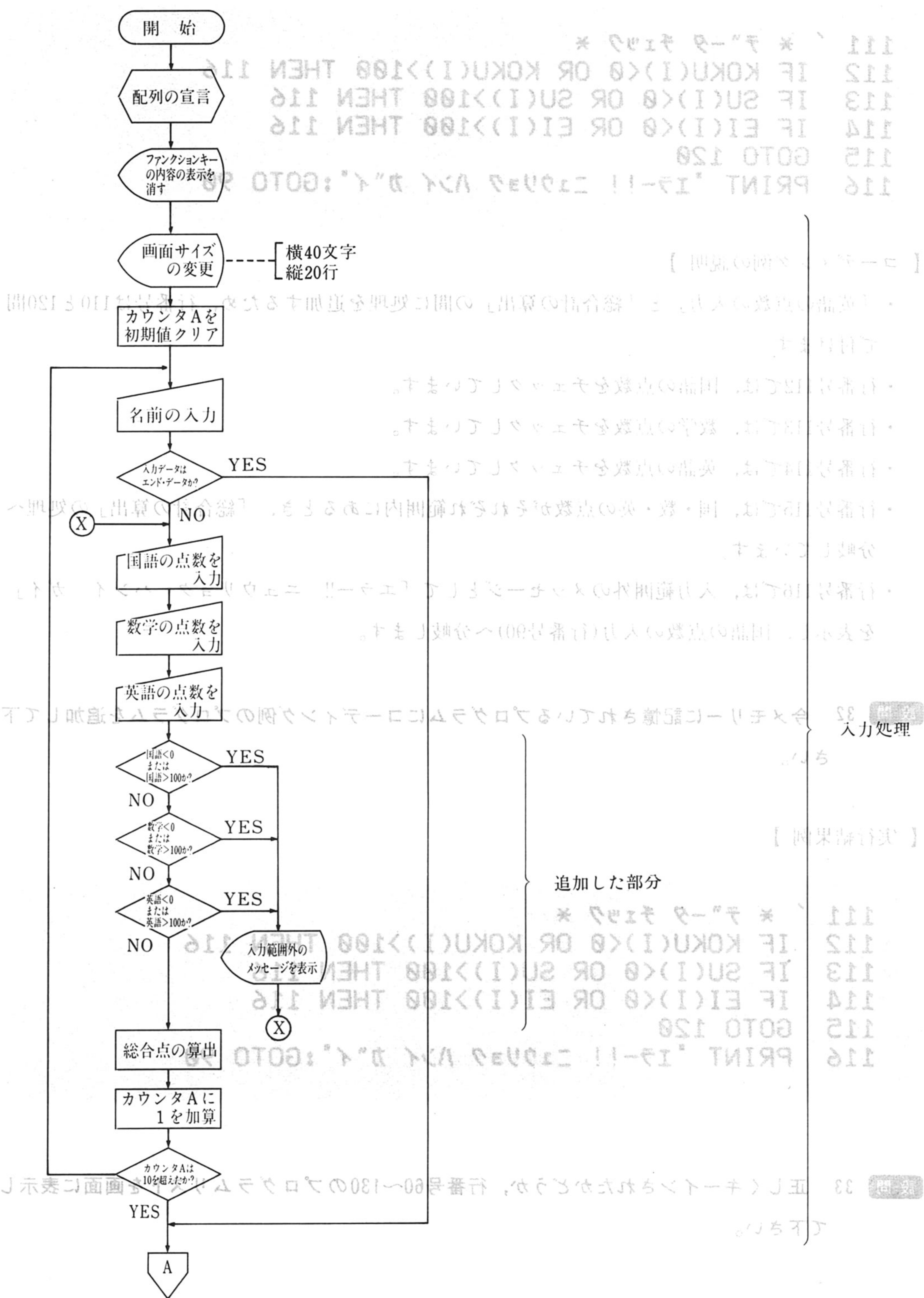
追加した部分

総合点の算出

カウンタAに1を加算

カウンタAは10を超えたか?　YES

入力処理

A

次に，追加したフローチャートからコーディングを行います。

【 コーディング例 】

```
111  ' * ﾃﾞｰﾀ ﾁｪｯｸ *
112  IF KOKU(I)<0 OR KOKU(I)>100 THEN 116
113  IF SU(I)<0 OR SU(I)>100 THEN 116
114  IF EI(I)<0 OR EI(I)>100 THEN 116
115  GOTO 120
116  PRINT "ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ":GOTO 90
```

【 コーディング例の説明 】

- 「英語の点数の入力」と「総合計の算出」の間に処理を追加するため，行番号は110と120間で付けます。
- 行番号112では，国語の点数をチェックしています。
- 行番号113では，数学の点数をチェックしています。
- 行番号114では，英語の点数をチェックしています。
- 行番号115では，国・数・英の点数がそれぞれ範囲内にあるとき，「総合計の算出」の処理へ分岐しています。
- 行番号116では，入力範囲外のメッセージとして「エラー!! ニュウリョク ハンイ ガイ」を表示し，国語の点数の入力(行番号90)へ分岐します。

設問 32 今メモリーに記憶されているプログラムにコーディング例のプログラムを追加して下さい。

【 実行結果例 】

```
111  ' * ﾃﾞｰﾀ ﾁｪｯｸ *
112  IF KOKU(I)<0 OR KOKU(I)>100 THEN 116
113  IF SU(I)<0 OR SU(I)>100 THEN 116
114  IF EI(I)<0 OR EI(I)>100 THEN 116
115  GOTO 120
116  PRINT "ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ":GOTO 90
```

設問 33 正しくキーインされたかどうか，行番号60〜130のプログラムリストを画面に表示して下さい。

【 実行結果例 】

```
list 60-130
60 FOR I=1 TO N
70   INPUT "ﾅﾏｴ (END=[E]) ";NAMAE$(I)
80   IF NAMAE$(I)="E" OR NAMAE$(I)="e" THEN *SORT.R
90   INPUT "ｺｸｺﾞ  = ",KOKU(I)
100  INPUT "ｽｳｶﾞｸ = ",SU(I)
110  INPUT "ｴｲｺﾞ  = ",EI(I):PRINT
111 ' * ﾃﾞｰﾀ ﾁｪｯｸ *
112  IF KOKU(I)<0 OR KOKU(I)>100 THEN 116
113  IF SU(I)<0 OR SU(I)>100 THEN 116
114  IF EI(I)<0 OR EI(I)>100 THEN 116
115  GOTO 120
116  PRINT "ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ":GOTO 90
120  SOGO(I)=KOKU(I)+SU(I)+EI(I)
130 NEXT I
Ok
```

なお，修正後のプログラムリストは次のようになります。

【 修正後のプログラムリスト 】

```
10 ' *** ｾｲｾｷｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ***
20 N=10:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
30 CONSOLE ,,0
40 ' ** ﾆｭｳﾘｮｸ ﾙｰﾁﾝ **
50 WIDTH 40,20
60 FOR I=1 TO N
70   INPUT "ﾅﾏｴ (END=[E]) ";NAMAE$(I)
80   IF NAMAE$(I)="E" OR NAMAE$(I)="e" THEN *SORT.RTN
90   INPUT "ｺｸｺﾞ  = ",KOKU(I)
100  INPUT "ｽｳｶﾞｸ = ",SU(I)
110  INPUT "ｴｲｺﾞ  = ",EI(I):PRINT
111 ' * ﾃﾞｰﾀ ﾁｪｯｸ *
112  IF KOKU(I)<0 OR KOKU(I)>100 THEN 116
113  IF SU(I)<0 OR SU(I)>100 THEN 116
114  IF EI(I)<0 OR EI(I)>100 THEN 116
115  GOTO 120
116  PRINT "ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ":GOTO 90
120  SOGO(I)=KOKU(I)+SU(I)+EI(I)
130 NEXT I
140 ' ** ｿｰﾄ ﾙｰﾁﾝ **
150 *SORT.RTN:I=I-1
160 FOR J1=1 TO I-1
170   MEMO=J1
180   FOR J2=J1+1 TO I
190     IF SOGO(MEMO)<SOGO(J2) THEN MEMO=J2
200   NEXT J2
```

（つづく）

```
210   SWAP NAMAE$(J1),NAMAE$(MEMO)
220   SWAP KOKU(J1),KOKU(MEMO)
230   SWAP SU(J1),SU(MEMO)
240   SWAP EI(J1),EI(MEMO)
250   SWAP SOGO(J1),SOGO(MEMO)
260 NEXT J1
270 ' ** ｼｭﾂﾘｮｸ ﾙｰﾁﾝ **
280 WIDTH 80,20
290 PRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 PRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 PRINT
320 FOR J=1 TO I
330   PRINT J;TAB(8);NAMAE$(J);TAB(22);
340   PRINT KOKU(J);TAB(30);SU(J);TAB(38);EI(J);TAB(46);SOGO(J)
350 NEXT J
360 CONSOLE ,,1:END
```

（2） プログラムの実行

設問 34 修正したプログラムを入力範囲外のデータを使って実行して下さい。

【 実行結果例 】

（入力画面）

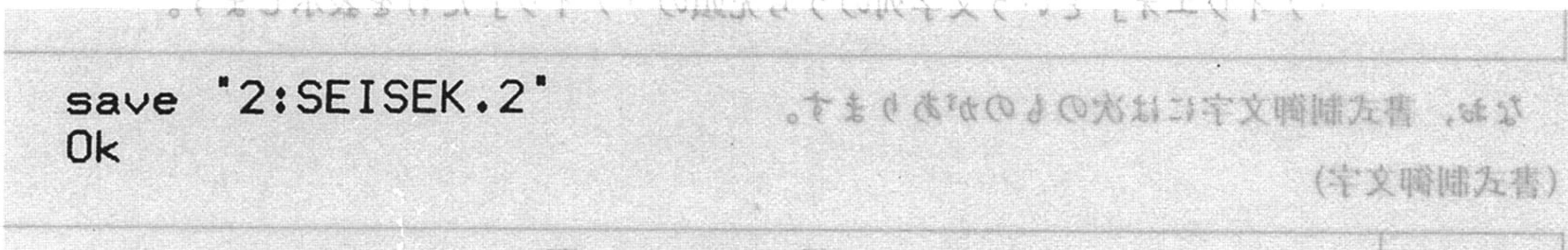
```
ﾅﾏｴ (END=[E]) ? ｻｯﾎﾟﾛ ｲﾁﾛｳ
ｺｸｺﾞ   = 80
ｽｳｶﾞｸ = 60
ｴｲｺﾞ   = 655

ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ
ｺｸｺﾞ   = -80
ｽｳｶﾞｸ = 60
ｴｲｺﾞ   = 65

ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ
ｺｸｺﾞ   = 80
ｽｳｶﾞｸ = 60
ｴｲｺﾞ   = 65

ﾅﾏｴ (END=[E]) ? ■
```

【 実行後の説明 】

・0点未満の点数か100点を超える点数が入力されると，「エラー!! ニュウリョク ハンイガイ」というメッセージを表示して，国・数・英の点数を入力し直しています。

設問 35　今メモリーに記憶されているプログラムを「SEISEK.2」という名前でドライブ2のフロッピーディスクへセーブして下さい。

【 実行結果例 】

```
save "2:SEISEK.2"
Ok
```

4.2.2 データを編集して表示する

また，出力されるデータはすべて左揃えになっていて，特に数値の場合は桁が違うと見づらくなります。そこで，データを編集(桁揃え，文字数の制限など)して表示します。

(1) プログラム修正の条件

プログラムを修正する条件は次のとおりです。

【 修正の条件 】

・明細行を編集して表示します。

・順位と国・数・英それぞれの点数，および総合点を右揃えで表示します。

・名前の表示は10桁以内とします。

(2) 必要な命令の説明

データを編集して表示するには「PRINT USING」命令を使います。PRINT USING命令の一般形式と書き方例は次のとおりです。

(PRINT USING命令の説明)

<一般形式>

PRINT USING　"書式制御文字列"；式

・文字列や数値を指定した書式で編集して画面に表示します。

・書式制御文字列には，編集する形式を指定します。

・式には，出力する数値データや文字列データを変数や定数で指定します。

<書き方例>

① PRINT USING "###,###"；12345

画面に「12345」という数値を編集して「12,345」と表示します。

② PRINT USING "￥￥####"；123

画面に「123」という数値を編集して「￥123」と表示します。

③ PRINT USING "&　&"；"アイウエオ"

書式制御文字列が「&　&」と3文字分を出力することを表しているので，画面に「アイウエオ」という文字列のうち先頭の「アイウ」だけを表示します。

なお，書式制御文字には次のものがあります。

(書式制御文字)

記号	説明
#	数値を出力する桁を指定します。出力される数値は右揃えになります。数値が負のときは負符号を数値の先頭に表示します。
.	小数点の位置を指定します。
+	数値の符号(「＋」または「－」)が数値の先頭または後に出力されます。書式制御文字列の前または後に付けます。
－	数式が負のときだけ「－」符号が出力されます。負符号を数値の後に表示したいときにだけ書式制御文字列の後に付けます。
＊＊	数値が小さくて左側に空白ができたとき，空白部分を「＊」で埋めます。書式制御文字列の先頭に付けます。
￥￥	数値の直前に「￥」記号を出力します。書式制御文字列の先頭に付けます。
＊＊￥	数値の直前に「￥」記号を出力し，左側の空白部分を「＊」で埋めます。書式制御文字列の先頭に付けます。
,	数値を3桁ごとに区切って「,」を出力します。書式制御文字列の先頭の「#」記号と「.」記号の間に指定します。
^ ^ ^ ^	数値を指数形式で出力します。「#」記号の後に付けます。
！	式(文字列)の最初の一文字だけ出力します。
&～&	文字列を左揃で出力します。式(文字列)の長さが書式制御文字列よりも長いときは，余分な文字は出力されません。

【 注意点 】

・書式制御文字列に制御文字(編集記号)以外の文字を指定したとき，その文字はそのまま出力されます。

・指定した書式制御文字列のうちの数値領域よりも式で与えた数値の桁数の方が大きいときは，数値の前に「%」が印字されます。

(3) プログラムの修正

修正の条件に従ってプログラムを修正します。明細を表示する命令文(行番号330～340)をPRINT USING命令を使った文にします。

【 コーディング例 】

```
330    PRINT USING " ##";J;
331    PRINT TAB(8);
332    PRINT USING "&          &";NAMAE$(J);
333    PRINT TAB(22);
334    PRINT USING "###";KOKU(J);
335    PRINT TAB(30);
336    PRINT USING "###";SU(J);
337    PRINT TAB(38);
338    PRINT USING "###";EI(J);
339    PRINT TAB(47);
340    PRINT USING "###";SOGO(J)
```

【 コーディング例の説明 】

・行番号332の書式制御文字列は「&」から「&」まで10文字分で，10文字以内の文字列を表示しています。

・行番号330では，2桁の数値を，行番号336,338,340では3桁の数値を表示しています。

・行番号330～339で命令文を「；」で終わっているのは，明細を1行に表示するため命令文を実行しても改行しないようにしているからです。

設問 36　今メモリーに記憶されているプログラムをコーディング例に従って修正して下さい。

設問 37　正しく修正できたかどうか，行番号320以降のプログラムリストを画面に表示して確認して下さい。

【 プログラムリスト 】

```
list 320-
320 FOR J=1 TO I
330   PRINT USING " ##";J;
331   PRINT TAB(8);
332   PRINT USING "&          &";NAMAE$(J);
333   PRINT TAB(22);
334   PRINT USING "###";KOKU(J);
335   PRINT TAB(30);
336   PRINT USING "###";SU(J);
```

(つづく)

```
337   PRINT TAB(38);
338   PRINT USING "###";EI(J);
339   PRINT TAB(47);
340   PRINT USING "###";SOGO(J)
350 NEXT J
360 CONSOLE ,,1:END
Ok
```

なお，修正後のプログラム全体のリストは次のようになります。

【 修正後のプログラムリスト 】

```
10 ' *** ｾｲｾｷｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ***
20 N=10:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
30 CONSOLE ,,0
40 ' ** ﾆｭｳﾘｮｸ ﾙｰﾁﾝ **
50 WIDTH 40,20
60 FOR I=1 TO N
70   INPUT "ﾅﾏｴ (END=[E]) ";NAMAE$(I)
80   IF NAMAE$(I)="E" OR NAMAE$(I)="e" THEN *SORT.RTN
90   INPUT "ｺｸｺﾞ  = ",KOKU(I)
100  INPUT "ｽｳｶﾞｸ = ",SU(I)
110  INPUT "ｴｲｺﾞ  = ",EI(I):PRINT
111 ' * ﾃﾞｰﾀ ﾁｪｯｸ *
112  IF KOKU(I)<0 OR KOKU(I)>100 THEN 116
113  IF SU(I)<0 OR SU(I)>100 THEN 116
114  IF EI(I)<0 OR EI(I)>100 THEN 116
115  GOTO 120
116  PRINT "ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ":GOTO 90
120  SOGO(I)=KOKU(I)+SU(I)+EI(I)
130 NEXT I
140 ' ** ｿｰﾄ ﾙｰﾁﾝ **
150 *SORT.RTN:I=I-1
160 FOR J1=1 TO I-1
170   MEMO=J1
180   FOR J2=J1+1 TO I
190     IF SOGO(MEMO)<SOGO(J2) THEN MEMO=J2
200   NEXT J2
210  SWAP NAMAE$(J1),NAMAE$(MEMO)
220  SWAP KOKU(J1),KOKU(MEMO)
230  SWAP SU(J1),SU(MEMO)
240  SWAP EI(J1),EI(MEMO)
250  SWAP SOGO(J1),SOGO(MEMO)
260 NEXT J1
270 ' ** ｼｭﾂﾘｮｸ ﾙｰﾁﾝ **
280 WIDTH 80,20
290 PRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 PRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 PRINT
320 FOR J=1 TO I
330   PRINT USING " ##";J;
331   PRINT TAB(8);
332   PRINT USING "&            &";NAMAE$(J);
333   PRINT TAB(22);
334   PRINT USING "###";KOKU(J);
```

（つづく）

```
335   PRINT TAB(30);
336   PRINT USING "###";SU(J);
337   PRINT TAB(38);
338   PRINT USING "###";EI(J);
339   PRINT TAB(47);
340   PRINT USING "###";SOGO(J)
350 NEXT J
360 CONSOLE ,,1:END
```

(4) プログラムの実行

設問 38 正しく修正できていれば，プログラムを次のデータで実行して下さい。

(入力するデータ)

名　　前	国語	数学	英語
サッポロ　イチロウ	80	60	65
トウキョウ　ヒロシ	35	5	50
ナゴヤ　カズコ	75	72	68
オオサカ　タツヤ	8	90	44
フクオカ　ユウタ	60	60	71
カゴシマ　ミユキ	65	85	70
E			

【 実行結果例 】

(出力画面)

```
ｼﾞｭﾝｲ   ﾅﾏｴ              ｺｸｺﾞ    ｽｳｶﾞｸ    ｴｲｺﾞ    ｿｳｺﾞｳ

  1     ｶｺﾞｼﾏ ﾐﾕｷ         65       85       70      220
  2     ﾅｺﾞﾔ ｶｽﾞｺ         75       72       68      215
  3     ｻｯﾎﾟﾛ ｲﾁﾛｳ        80       60       65      205
  4     ﾌｸｵｶ ﾕｳﾀ          60       60       71      191
  5     ｵｵｻｶ ﾀﾂﾔ           8       90       44      142
  6     ﾄｳｷｮｳ ﾋﾛｼ         35        5       50       90
Ok
```

設問 39 正しく実行できていたら，このプログラムを「SEISEK.3」という名前でドライブ2のフロッピーディスクへセーブして下さい。

【 実行結果例 】

```
save "2:SEISEK.3"
Ok
```

4.3　結果をプリンタに印字する

今まで作成してきた成績処理プログラムは，10件分までのデータしか処理できません。これを，もっと大量のデータでも処理できるように変更したいのですが，処理した結果を画面に表示すると最大25件分しか出力できません。それに，画面へ表示したデータは一時的なもので，電源を切ると消えてしまいます。

そこで，処理した結果をプリンタに印字するようにします。そうすると，出力した結果は残りますし，用紙が切れない限り100件でも200件でもデータを出力することができます。

この項では，処理結果のプリンタへの印字と，プリンタの拡大文字および縮小文字の使い方について学習します。

プログラムを修正する条件は次のとおりです。

【 修正の条件 】

・データを最大50件まで処理できるようにします。

・画面に表示していた処理結果をプリンタへ印字するようにします。

・見出しを項目見出しの前に拡大文字で印字します。

・明細行は縮小文字で印字します。

・印字の形式(フォーマット)は次に示すとおりです。

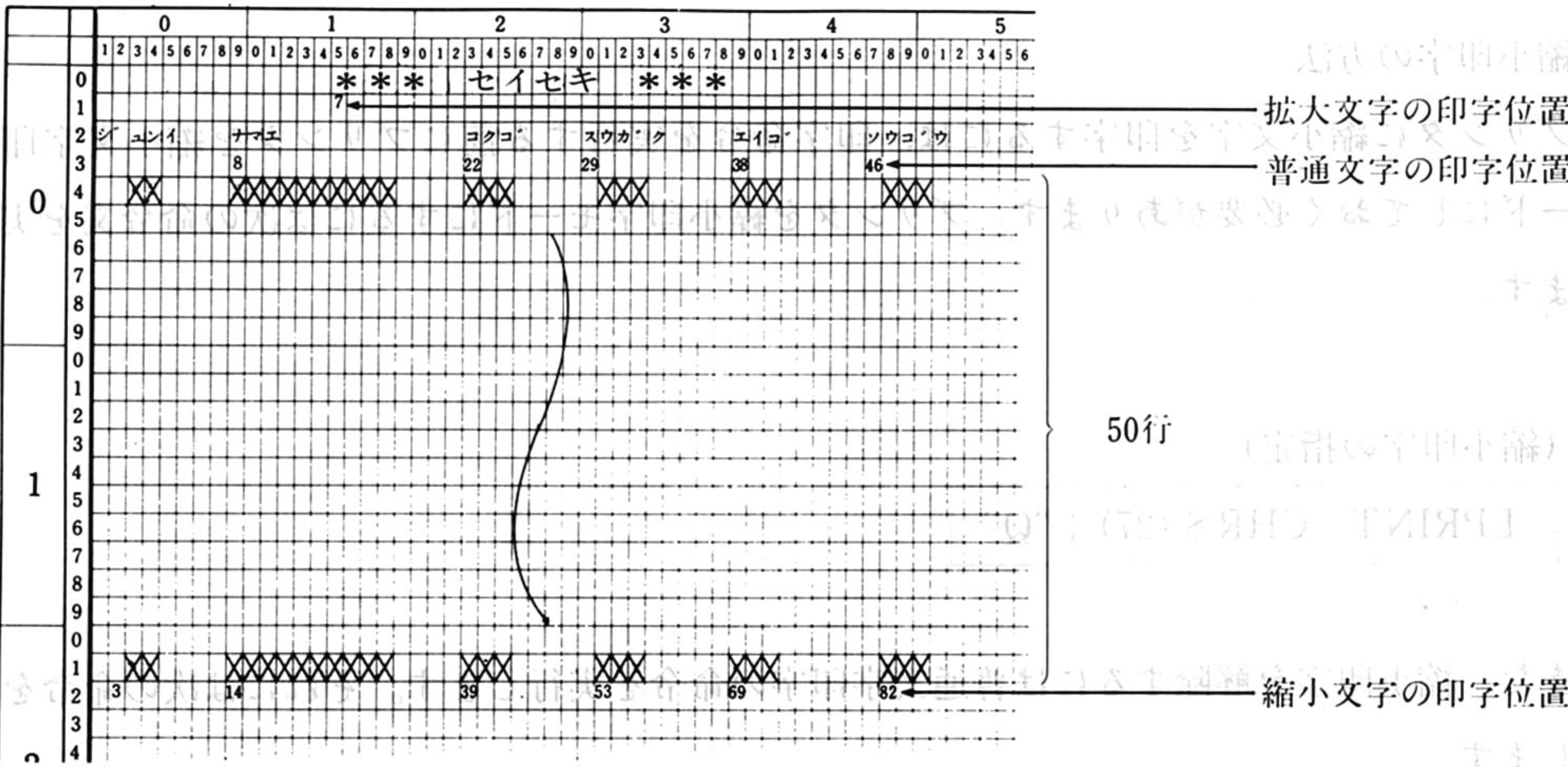
＊＊＊　セイセキ　＊＊＊

ジュンイ　ナマエ　コクゴ　スウガク　エイゴ　ソウゴウ

拡大文字の印字位置

普通文字の印字位置

50行

縮小文字の印字位置

4.3.1　拡大印字と縮小印字

（1）　拡大印字の方法

プリンタに拡大文字を印字するには，印字命令を実行する前にプリンタを拡大文字印字モードにしておく必要があります。プリンタを拡大印字モードにするには次の命令文を実行します。

（拡大印字の指定）

```
LPRINT   CHR＄(14)
```

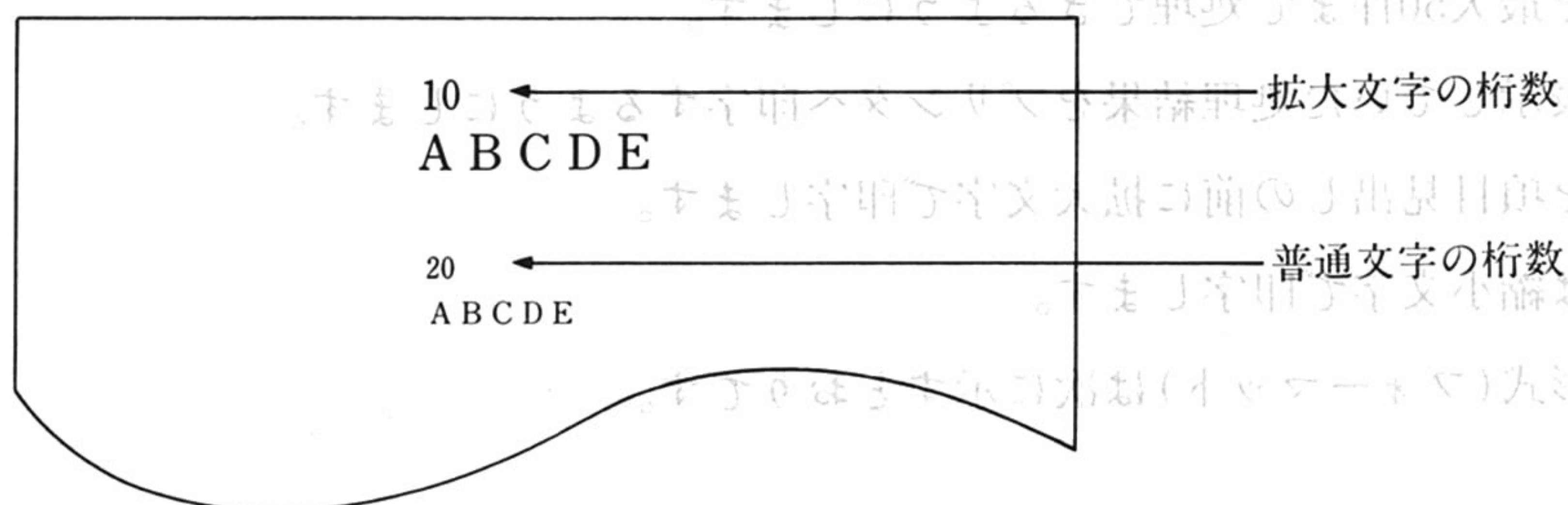
また，拡大印字を解除するには次の命令文を実行します。

（拡大印字解除の指定）

```
LPRINT   CHR＄(15)
```

拡大印字を指定してから解除するまでの間は，プリンタには拡大文字が印字されます。拡大文字は横幅が普通文字のちょうど2倍になるため，1行に印字できる文字数は普通文字の半分の40文字になります。そのため，TAB関数などで印字位置を指定するときは特に注意が必要です。

10 ← 拡大文字の桁数
ABCDE

20 ← 普通文字の桁数
ABCDE

（２） 縮小印字の方法

プリンタに縮小文字を印字するには，印字命令を実行する前にプリンタを縮小文字印字モードにしておく必要があります。プリンタを縮小印字モードにするには次の命令文を実行します。

（縮小印字の指定）

```
LPRINT   CHR＄(27)；"Q"
```

また，縮小印字を解除するには普通文字印字の命令を実行します。それには次の命令を実行します。

（普通文字印字の指定）

```
LPRINT   CHR＄(27)；"H"
```

縮小印字モードにすると，1行に印字できる文字数は136文字になります。

（3） 縮小の拡大文字

縮小印字の指定と拡大印字の指定を組み合わせて「縮小の拡大文字」を印字することができます。縮小の拡大文字印字の指定は次のようにします。

（縮小の拡大文字印字の指定）

```
LPRINT  CHR$(27);"Q";
LPRINT  CHR$(14)
```

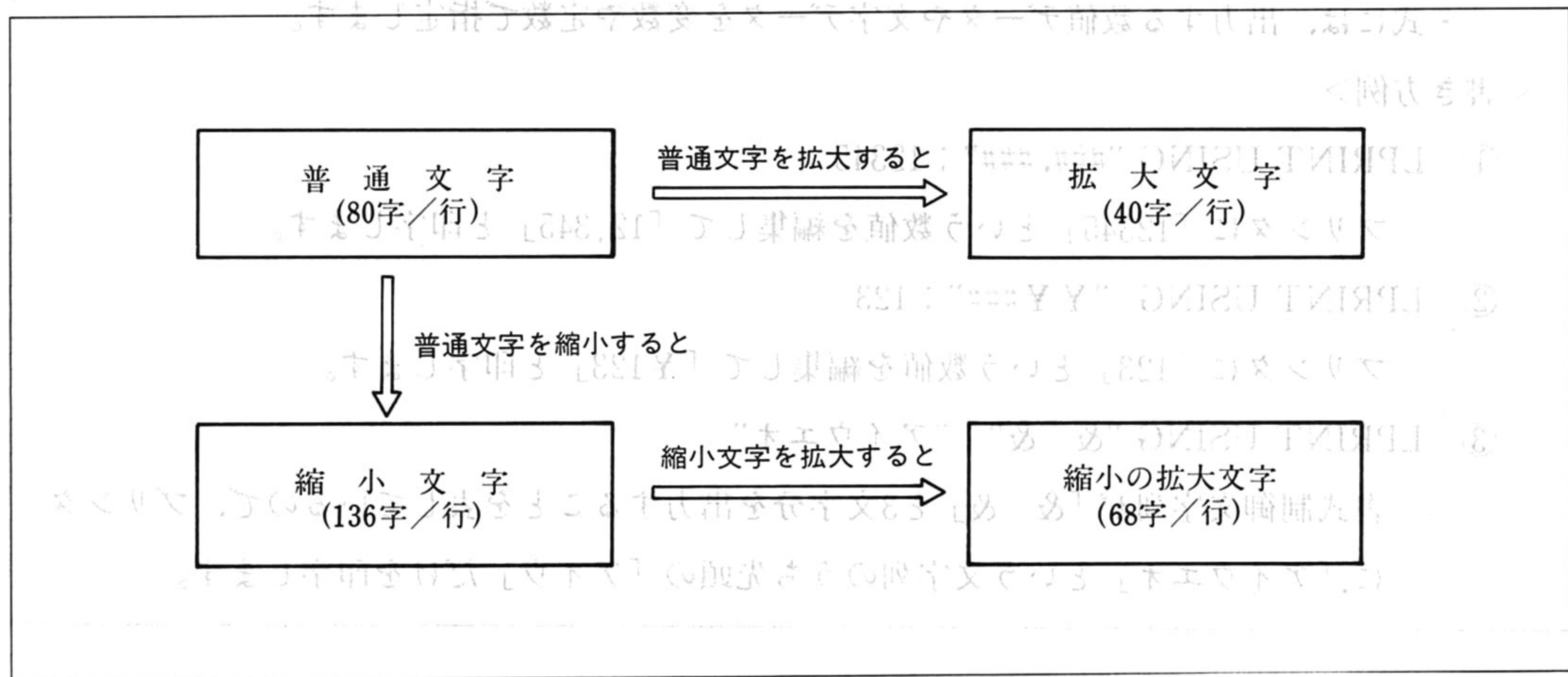
また，解除は次のようにします。

（縮小の拡大文字印字の解除）

```
LPRINT  CHR$(15);
LPRINT  CHR$(27);"H"
```

縮小の拡大文字印字にすると，1行に印字できる文字数は68文字になります。

これらのことを図にまとめると，次のようになります。

普 通 文 字 (80字／行)	普通文字を拡大すると ⇒	拡 大 文 字 (40字／行)
⇓ 普通文字を縮小すると		
縮 小 文 字 (136字／行)	縮小文字を拡大すると ⇒	縮小の拡大文字 (68字／行)

4.3.2 プログラムの修正

今メモリーに記憶されているプログラムを【 修正の条件 】に従って，次の手順で修正することにします。

（1） データの処理件数を最大10件から50件にする

データの最大処理件数は，行番号20の「N＝50」で指定しています。そこで，行番号20をつぎのように変更します。

```
20 N=10:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
```

⇩

```
20 N=50:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
```

（2） 処理結果の出力を画面からプリンタに変更する

画面表示のPRINT命令やPRINT　USING命令をプリンタへ印字するLPRINT命令やLPRINT USING命令に変更します。

LPRINT USING命令の一般形式と書き方例は次のとおりです。

(LPRINT USING命令の説明)

<一般形式>

LPRINT USING "書式制御文字列"；式

・文字列や数値を指定した形式で編集してプリンタに印字します。

・書式制御文字列には，編集する形式を指定します。

・式には，出力する数値データや文字データを変数や定数で指定します。

<書き方例>

① LPRINT USING "###,###"；12345

プリンタに「12345」という数値を編集して「12,345」と印字します。

② LPRINT USING "￥￥###"；123

プリンタに「123」という数値を編集して「￥123」と印字します。

③ LPRINT USING "&　&"；"アイウエオ"

書式制御文字列が「&　&」と3文字分を出力することを表しているので，プリンタに「アイウエオ」という文字列のうち先頭の「アイウ」だけを印字します。

行番号290〜340のPRINT命令を次のように変更します。

```
290 PRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 PRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 PRINT
320 FOR J=1 TO I
330   PRINT USING " ##";J;
331   PRINT TAB(8);
332   PRINT USING "&          &";NAMAE$(J);
333   PRINT TAB(22);
334   PRINT USING "###";KOKU(J);
335   PRINT TAB(30);
```

（つづく）

```
336   PRINT USING "###";SU(J);
337   PRINT TAB(38);
338   PRINT USING "###";EI(J);
339   PRINT TAB(47);
340   PRINT USING "###";SOGO(J)
```

⇩

```
290 LPRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 LPRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 LPRINT CHR$(27);"Q"
320 FOR J=1 TO I
330   LPRINT USING " ##";J;
331   LPRINT TAB(8);
332   LPRINT USING "&         &";NAMAE$(J);
333   LPRINT TAB(22);
334   LPRINT USING "###";KOKU(J);
335   LPRINT TAB(30);
336   LPRINT USING "###";SU(J);
337   LPRINT TAB(38);
338   LPRINT USING "###";EI(J);
339   LPRINT TAB(47);
340   LPRINT USING "###";SOGO(J)
```

（3） 見出しを項目見出しの前に拡大文字で印字する

見出し「＊＊＊　セイセキ　＊＊＊」を拡大印字します。行番号280を次のように変更します。

```
280 WIDTH 80,20
```

⇩

```
280 LPRINT CHR$(14)
282 LPRINT TAB(7);"*** ｾｲｾｷ ***"
284 LPRINT CHR$(15)
```

処理結果は画面には表示しないため，「280　WIDTH 80,20」は不要になります。

（4） 明細行は縮小文字で印字する。

行番号320の前と行番号350の後に次の命令文を追加します。

```
310 LPRINT CHR$(27);"Q"
355 LPRINT CHR$(27);"H"
```

処理結果は画面には表示しないため，「310　PRINT」は不要になります。

（5） 印字フォーマットに従って印字位置を変更する

明細行を縮小印字するため，このままでは項目見出しと明細の位置が合わなくなります。そこで，行番号330～340の中のTAB関数の値を【 修正の条件 】で示したフォーマット（縮小文字の印字位置）に従って変更します。

```
330   LPRINT USING " ##";J;
331   LPRINT TAB(8);
332   LPRINT USING "&          &";NAMAE$(J);
333   LPRINT TAB(22);
334   LPRINT USING "###";KOKU(J);
335   LPRINT TAB(30);
336   LPRINT USING "###";SU(J);
337   LPRINT TAB(38);
338   LPRINT USING "###";EI(J);
339   LPRINT TAB(47);
340   LPRINT USING "###";SOGO(J)
```

⇩

```
330   LPRINT USING " ##";J;
331   LPRINT TAB(14);
332   LPRINT USING "&          &";NAMAE$(J);
333   LPRINT TAB(39);
334   LPRINT USING "###";KOKU(J);
335   LPRINT TAB(53);
336   LPRINT USING "###";SU(J);
337   LPRINT TAB(67);
338   LPRINT USING "###";EI(J);
339   LPRINT TAB(82);
340   LPRINT USING "###";SOGO(J)
```

さらに，次の命令文を追加します。

```
329   LPRINT TAB(3);
```

設問 40 今メモリーに記憶されているプログラムを前ページまでの手順に従って修正して下さい。

設問 41 正しく修正できているかどうか，プリンタにすべてのプログラムリストを印字して下さい。

【 プログラムリスト印字例 】

```
10 ' *** ｾｲｾｷｼｮﾘ ﾌﾟﾛｸﾞﾗﾑ ***
20 N=50:DIM NAMAE$(N),KOKU(N),SU(N),EI(N),SOGO(N)
30 CONSOLE ,,0
40 ' ** ﾆｭｳﾘｮｸ ﾙｰﾁﾝ **
50 WIDTH 40,20
60 FOR I=1 TO N
70   INPUT "ﾅﾏｴ (END=[E]) ";NAMAE$(I)
80   IF NAMAE$(I)="E" OR NAMAE$(I)="e" THEN *SORT.RTN
90   INPUT "ｺｸｺﾞ  = ",KOKU(I)
100  INPUT "ｽｳｶﾞｸ = ",SU(I)
110  INPUT "ｴｲｺﾞ  = ",EI(I):PRINT
111 ' * ﾃﾞｰﾀ ﾁｪｯｸ *
112  IF KOKU(I)<0 OR KOKU(I)>100 THEN 116
113  IF SU(I)<0 OR SU(I)>100 THEN 116
114  IF EI(I)<0 OR EI(I)>100 THEN 116
115  GOTO 120
116  PRINT "ｴﾗｰ!! ﾆｭｳﾘｮｸ ﾊﾝｲ ｶﾞｲ":GOTO 90
120  SOGO(I)=KOKU(I)+SU(I)+EI(I)
130 NEXT I
140 ' ** ｿｰﾄ ﾙｰﾁﾝ **
150 *SORT.RTN:I=I-1
160 FOR J1=1 TO I-1
170   MEMO=J1
180   FOR J2=J1+1 TO I
190     IF SOGO(MEMO)<SOGO(J2) THEN MEMO=J2
200   NEXT J2
210  SWAP NAMAE$(J1),NAMAE$(MEMO)
220  SWAP KOKU(J1),KOKU(MEMO)
230  SWAP SU(J1),SU(MEMO)
240  SWAP EI(J1),EI(MEMO)
250  SWAP SOGO(J1),SOGO(MEMO)
260 NEXT J1
270 ' ** ｼｭﾂﾘｮｸ ﾙｰﾁﾝ **
280 LPRINT CHR$(14)
282 LPRINT TAB(7);"*** ｾｲｾｷ ***"
284 LPRINT CHR$(15)
290 LPRINT "ｼﾞｭﾝｲ";TAB(8);"ﾅﾏｴ";TAB(22);
300 LPRINT "ｺｸｺﾞ";TAB(29);"ｽｳｶﾞｸ";TAB(38);"ｴｲｺﾞ";TAB(46);"ｿｳｺﾞｳ"
310 LPRINT CHR$(27);"Q"
320 FOR J=1 TO I
329   LPRINT TAB(3);
330   LPRINT USING " ##";J;
331   LPRINT TAB(14);
332   LPRINT USING "&          &";NAMAE$(J);
333   LPRINT TAB(39);
334   LPRINT USING "###";KOKU(J);
335   LPRINT TAB(53);
336   LPRINT USING "###";SU(J);
```

（つづく）

```
337   LPRINT TAB(67);
338   LPRINT USING "###";EI(J);
339   LPRINT TAB(82);
340   LPRINT USING "###";SOGO(J)
350 NEXT J
355 LPRINT CHR$(27);"H"
360 CONSOLE ,,1:END
```

4.3.3 プログラムの実行

設問 42　正しく修正できていたら，次のデータを使ってプログラムを実行して下さい。

(入力するデータ)

	名　前	国語	数学	英語
1	サッポロ　イチロウ	80	60	65
2	アキタ　ヒデオ	50	85	55
3	センダイ　ミツル	60	50	45
4	ヤマガタ　ナオミ	74	63	88
5	フクシマ　ヒロアキ	82	98	80
6	ミト　ケイイチ	71	54	93
7	マエバシ　クミコ	77	64	84
8	チバ　ヤスヒロ	53	43	49
9	トウキョウ　ヒロシ	35	5	50
10	ヨコハマ　アサミ	48	61	58
11	ナガノ　カズヨシ	19	48	12
12	ニイガタ　ユキオ	56	9	34
13	カナザワ　ショウコ	80	72	96
14	フクイ　チエミ	73	100	61
15	ナゴヤ　カズコ	75	72	68
16	オオツ　トシユキ	60	100	53
17	キョウト　エリコ	74	51	54
18	オオサカ　タツヤ	8	90	44
19	カンベ　アキラ	79	24	100
20	ナラ　ケイコ	92	38	56
21	マツエ　コウジ	68	100	51
22	オカヤマ　ユウコ	91	70	60

	名　　前		国語	数学	英語
23	ヒロシマ	ユキオ	47	88	30
24	ヤマグチ	ミチヨ	95	51	75
25	タカマツ	シンヤ	84	82	100
26	マツヤマ	トシエ	86	50	62
27	フクオカ	ユウタ	60	60	71
28	ナガサキ	リツコ	90	68	100
29	ミヤザキ	ツカサ	57	100	41
30	カゴシマ	ミユキ	65	85	70

(注)　データを全部入力し終わったら名前の代わりに「E」または「e」を入力します。

【 実行結果例 】

```
                    *** セイセキ ***

ｼﾞｭﾝｲ   ﾅﾏｴ              ｺｸｺﾞ   ｽｳｶﾞｸ   ｴｲｺﾞ   ｿｳｺﾞｳ

   1    ﾀｶﾏﾂ ｼﾝﾔ           84      82     100     266
   2    ﾌｸｼﾏ ﾋﾛｱｷ          82      98      80     260
   3    ﾅｶﾞｻｷ ﾘﾂｺ          90      68     100     258
   4    ｶﾅｻﾞﾜ ｼｮｳｺ         80      72      96     248
   5    ﾌｸｲ ﾁｴﾐ            73     100      61     234
   6    ﾏｴﾊﾞｼ ｸﾐｺ          77      64      84     225
   7    ﾔﾏｶﾞﾀ ﾅｵﾐ          74      63      88     225
   8    ｵｶﾔﾏ ﾕｳｺ           91      70      60     221
   9    ﾔﾏｸﾞﾁ ﾐﾁﾖ          95      51      75     221
  10    ｶｺﾞｼﾏ ﾐﾕｷ          65      85      70     220
  11    ﾏﾂｴ ｺｳｼﾞ           68     100      51     219
  12    ﾐﾄ ｹｲｲﾁ            71      54      93     218
  13    ﾅｺﾞﾔ ｶｽﾞｺ          75      72      68     215
  14    ｵｵﾂ ﾄｼﾕｷ           60     100      53     213
  15    ｻｯﾎﾟﾛ ｲﾁﾛｳ         80      60      65     205
  16    ｶﾝﾍﾞ ｱｷﾗ           79      24     100     203
```

(つづく)

```
17  ﾏﾂﾔﾏ ﾄｼｴ     86   50   62  198
18  ﾐﾔｻﾞｷ ﾂｶｻ    57  100   41  198
19  ﾌｸｵｶ ﾕｳﾀ     60   60   71  191
20  ｱｷﾀ ﾋﾃﾞｵ     50   85   55  190
21  ﾅﾗ ｹｲｺ       92   38   56  186
22  ｷｮｳﾄ ｴﾘｺ     74   51   54  179
23  ﾖｺﾊﾏ ｱｻﾐ     48   61   58  167
24  ﾋﾛｼﾏ ﾕｷｵ     47   88   30  165
25  ｾﾝﾀﾞｲ ﾐﾂﾙ    60   50   45  155
26  ﾁﾊﾞ ﾔｽﾋﾛ     53   43   49  145
27  ｵｵｻｶ ﾀﾂﾔ      8   90   44  142
28  ﾆｲｶﾞﾀ ﾕｷｵ    56    9   34   99
29  ﾄｳｷｮｳ ﾋﾛｼ    35    5   50   90
30  ﾅｶﾞﾉ ｶｽﾞﾖｼ   19   48   12   79
```

設問 43　正しく実行できていたら，今メモリーに記憶しているプログラムをドライブ2のフロッピーディスクへ「SEISEK.4」という名前でセーブして下さい。

【 実行結果例 】

```
save "2:SEISEK.4"
Ok
```

なお，フローチャートは最終的には次のようになります。

【 フローチャート例 】

開　始

配列の宣言

ファンクションキーの内容の表示を消す

A

準備処理

B

カウンタＩから
１を減算

カウンタJ1を
初期値クリア

ワーク変数MEMOに
カウンタJ1を代入

カウンタJ2を
初期値クリア

配列（総合上）
を比較

総合点(MEMO)＜総合点（Ｊ２）

ワークMEMOに
カウンタJ2を代入

総合点(MEMO)
≧総合点（Ｊ２）

カウンタJ2に
１を加算

カウンタJ2は
カウンタＩの値を
超えたか

NO

YES

配列中のデータ
を交換する

添字MEMO 添字カウンタJ1
配列要素の内容を交換

カウンタJ1に
１を加算

カウンタJ1は
カウンタＩの値－１を
超えたか？

NO

YES

C

並べ換え処理

C

見出しの印字 ―― 見出しは拡大文字で印字

項目見出しの印字 ―― 小見出しは普通文字で印字

プリンタを縮少印字モードにする

カウンタJを初期値クリア

明細（配列の内容）を印字

カウンタJに1を加算

カウンタJはカウンタIの値を超えたか？
NO → 明細（配列の内容）を印字 へ戻る
YES

プリンタを普通文字印字モードにする

（以上 出力処理）

ファンクションキーの内容を表示する

（後処理）

終了

4.3.4 その他の印字制御のいろいろ

これまで，拡大文字や縮小文字の印字制御について学習してきましたが,このほかにもプリンタ(PC-8821/PC-8822)は，さまざまな印字機能を持っています。

ここでは，プリンタが持つ印字機能のうち主なものを説明します。

（1） 印字文字タイプの指定方法

・ハイスピードパイカ印字指定 LPRINT　CHR＄(27)；"N"；
・プロポーショナル印字指定 LPRINT　CHR＄(27)；"P"；
・エリート印字指定 LPRINT　CHR＄(27)；"E"；
・強調文字指定 LPRINT　CHR＄(27)；"！"；
・強調文字解除指定 LPRINT　CHR＄(27)；CHR＄(34)；

上の命令を使用したプログラム例と実行結果例を次に示します。

【 プログラム例と実行結果例 】

印字文字タイプ区分	プログラム例	実行結果例
ハイスピードパイカ印字	10 LPRINT CHR$(27);"N"; 20 LPRINT "ABCDE" 30 LPRINT "12345"	ABCDE 12345
プロポーショナル印字	10 LPRINT CHR$(27);"P"; 20 LPRINT "ABCDE" 30 LPRINT "12345"	ABCDE 12345
エリート印字	10 LPRINT CHR$(27);"E"; 20 LPRINT "ABCDE" 30 LPRINT "12345"	ABCDE 12345
強調印字	10 LPRINT CHR$(27);"!"; 20 LPRINT "ABCDE" 30 LPRINT "12345" 40 LPRINT CHR$(27);CHR$(34);	ABCDE 12345

（2） 改行に対しての指定

印字した後，用紙を送ることを「プリンタの改行」といいます。プリンタの改行は普通6行/インチ間隔で行われますが，この改行幅を変更することができます。

また，数行を一度に改行したり，改ページして次の用紙の先頭から印字するようなこともできます。

・6行/インチ間隔での印字指定 LPRINT CHR$(27);"A";　　電源投入直後はこの状態です。
・8行/インチ間隔での印字指定 LPRINT CHR$(27);"B";
・n行分の改行指定 LPRINT CHR$(31);CHR$(16+n); →改行数で，0≦n≦15の範囲
・改ページ指定 LPRINT CHR$(31);CHR$(1);

上の命令を使用したプログラム例と実行結果例を次に示します。

【 プログラム例と実行結果例 】

改行区分	プログラム例	実行結果例
6行/インチ間隔での印字指定	10 LPRINT CHR$(27);"A"; 20 LPRINT "ABCDE" 30 LPRINT "12345"	ABCDE 12345
8行/インチ間隔での印字指定	10 LPRINT CHR$(27);"B"; 20 LPRINT "ABCDE" 30 LPRINT "12345"	ABCDE 12345
n行分の改行指定	10 LPRINT "ABCDE" 20 LPRINT CHR$(31);CHR$(16+2); 30 LPRINT "12345"	ABCDE 12345
改ページ指定	10 LPRINT CHR$(31);CHR$(1);	この命令文が実行されるとＴＯＦの位置まで用紙が空送りされます。

（3） その他の指定

そのほかにも，次のような制御指定があります。

・アンダーライン印字指定 LPRINT CHR$(27);"X";

・アンダーライン印字解除の指定 LPRINT　CHR＄(27)；"Y"；
・レフトマージンの指定 LPRINT　CHR＄(27)；"L"；"n"；　　左からn文字分の空白を取ります。 nは3桁の10進数(例，010)
・ひらがな印字の指定 LPRINT　CHR＄(27)；"&";
・ひらがな印字解除の指定 LPRINT　CHR＄(27)；"＄"；
・順方向改行(用紙送り)の指定 LPRINT　CHR＄(27)；"f"；　　電源投入直後はこの状態。
・逆方向改行(用紙送り)の指定 LPRINT　CHR＄(27)；"r"；

プログラム例と実行結果例は次のとおりです。

【 プログラム例と実行結果例 】

項　　目	プログラム例	実行結果例
アンダーライン印字	10 LPRINT CHR$(27);"X"; 20 LPRINT "ABCDE" 30 LPRINT "12345" 40 LPRINT CHR$(27);"Y";	ABCDE 12345
レフトマージン	10 LPRINT CHR$(27);"L";"010"; 20 LPRINT "ABCDE" 30 LPRINT "12345"	ABCDE 12345 10文字分空白
ひらがな印字	10 LPRINT "ｱｲｳｴｵ" 20 LPRINT CHR$(27);"&"; 30 LPRINT "ｱｲｳｴｵ" 40 LPRINT CHR$(27);"$";	アイウエオ あいうえお
順方向改行	10 LPRINT CHR$(27);"f";	印字後の改行を順方向にセットします。
逆方向改行	20 LPRINT CHR$(27);"r";	印字後の改行を逆方向にセットします。

5章　基本的な処理のテクニック(2)

前章ではソートについて学習しましたが，この章では，もう一つの基本的な処理テクニックとして「検索(サーチ，Search)」を学習します。サーチは「照合」，「更新」，「抽出」などの処理テクニックに応用できます。

5.1　プログラム例(4) ―― 売上集計プログラム

今までは，配列の内容をすべて画面やプリンタに出力してきましたが，今回は，条件に合った特定の配列要素を探し出し，その内容を出力したり計算したりする処理を学習します。

この項では，電気店のレジをモデルにした売上集計プログラムを作成してみることにしましょう。

5.1.1　プログラムに必要な条件

プログラムを作成するための条件は次のとおりです。

【 作成の条件 】

・商品コードと数量をキーインして商品コード，商品名，数量，金額をプリンタに印字します。

・商品コードと商品名および単価は，あらかじめ配列に記憶させておきます。

・配列に記憶させる商品のデータは10件とし，その商品コード，商品名および単価は次のものを使います。

商品コード	商品名	単　価
29	ビデオ	183000
12	クーラー	109800
16	ドライヤー	7200
9	トースター	8800
3	クーリナー	16500
28	ステレオ	222000
10	テレビ	79000
21	ラジカセ	58000
7	レイゾウコ	137500
6	デンシレンジ	123000

・商品コードをキーインしたら，商品コードをキーにして商品名と単価をサーチします。

・配列にない商品コードを入力したときは，もう一度商品コードと単価を入力し直します。

・商品名と単価がサーチできたら，数量をキーインします。

・商品コードと数量を入力したら，商品コード，商品名，単価，数量を画面に表示して確認します。

・確認して，入力したデータが正しければ「Y」を入力し，金額と売上合計を計算し，明細を印字します。

・確認して，入力したデータが間違っていれば「N」を入力し，商品コードと数量を入力し直します。

・商品コードに「999」を入力したら，売上合計を印字します。

・印字フォーマットは次のとおりです。

```
0 :        *****ウリアケ゛ メイサイ*****
1 :
2 : ##  XXXXXXXXXX   ###,###   ¥###,###,###
     ︙
17: ##  XXXXXXXXXX   ###,###   ¥###,###,###
18:
19:              コ゛ウケイ   ¥##,###,###,###
                 14       22
```

5.1.2 フローチャートを作る

まず，フローチャートを作ります。

【 フローチャート例 】

開　始
配列宣言 ‐‐ 商品コード，商品名
単価
画面クリア
見出し印字
カウンタを
初期値クリア
配列に
データをセット ‐‐ プログラム中に
データを定義
カウンタに
1を加算
カウンタは
10を超えたか?
NO
YES
準備処理
商品コードの入力
商品コードは
「999」か?
YES
売上合計の印字
終　了
終了処理
NO
商品名，単価を
配列からサーチ
配列中に
入力した商品コード
はあるか?
存在
しない
存在する
数量の入力
確認のための
データ表示 ‐‐ 商品コード，商品名
単価，数量
確認の入力 ‐‐ 「Y」または「N」を入力
小文字も可
確認は
OKか?
NO
YES
金額と売上合計
を計算
入力処理
明細の印字
出力処理

5.1.3　プログラムに必要な命令の説明

売上集計プログラムに新たに必要となる命令および関数を次に説明します。

（1）　データの定義と読み込み

定められたデータをあらかじめ配列などに定義しておくのに，いちいち「＝」を使って代入したりINPUT命令でキーボードから入力したりすると，非常に面倒です。そこで，このような場合に「DATA」命令や「READ」命令を使うと，プログラムを実行したときに自動的に配列にデータが定義されるので便利です。

データを定義するDATA命令の一般形式は次のとおりです。

（DATA命令の説明）

<一般形式>

DATA　定数，定数，……，定数

・READ命令で読み取られる数値や文字をプログラム中に定義します。

・DATA命令は非実行文なので，プログラムのどこに書いてもかまいません。

また，データを読み込むREAD命令の一般形式は次のとおりです。

（READ命令の説明）

<一般形式>

READ　変数，変数，……，変数

・DATA命令で定義されている定数を指定した変数に読み込みます。

DATA命令とREAD命令はそれぞれ単独に使用することはありません。READ命令を使ったプログラム中には，必ずDATA命令でデータを定義しておかなければなりません。

DATA命令とREAD命令の書き方例を次に示します。

<書き方例>

①
```
10 DATA 10, 20, 80
20 DATA 350, 450
   :
50 READ A, B, C, D, E
```

変数A～Eに「10」～「450」のデータを次のように読み込みます。

変　数	A	B	C	D	E
内　容	10	20	80	350	45

②
```
10 DATA "AA","BB", 2390
   :
80 READ X$, Y$, Z%
```

変数X＄，Y＄に「AA」と「BB」を，変数Z%に「2390」をそれぞれ読み込みます。

変　数	X＄	Y＄	Z%
内　容	AA	BB	2390

(2) データを読み込む位置の変更

今回のプログラムでは使いませんが，PC-8801mkⅡではREAD命令で読むDATA文の位置を指定することもできます。データを読み込む位置を指定するには「RESTORE」命令を使います。 RESTORE命令の一般形式と書き方例は次のとおりです。

(RESTORE命令の説明)

<一般形式>

RESTORE　行番号

・READ命令で読み込むデータの開始位置を，指定した行番号のDATA文から読み込むように変更します。

・行番号の指定を省略すると，READ命令はプログラム中の最初に書かれているDATA文からデータを読み込みます。

<書き方例>

①
```
 10 DATA 100, 200
 20 DATA 400, 600
 30 DATA "XX", "YY"
      :
100 READ A, B
110 RESTORE 30
120 READ C$, D$
```

変数A，Bに「100」と「200」が、変数C$，D$に「XX」と「YY」が読み込まれます。

変 数	A	B	C$	D$
内 容	100	200	XX	YY

②
```
10 DATA "ABC","XYZ", 15, 45
      :
50 READ S$, T$, U, V
60 RESTORE
70 READ X$, Y$
```

変数S$～Vに「ABC」～「45」が，変数X$，Y$に「ABC」と「XYZ」が読み込まれます。

変 数	S$	T$	U	V	X$	Y$
内 容	ABC	XYZ	15	45	ABC	XYZ

(3) 配列からデータを探し出す。

PC-8801mkⅡには，配列中から指定されたデータを探す関数が用意してあります。データを探し出すには「SEARCH」関数を使います。SEARCH関数の一般形式と書き方例は次のとおりです。

(SEARCH関数の説明)

<一般形式>

変数＝SEARCH(配列名，整数値，開始番号，ステップ値)

・配列名で指定された配列の中から整数値と同じ値を探し出し，そのときの添字を変数に与えます。

・配列中に指定した整数値と同じ値が見つからなかったら，変数には「－1」が与えられます。

・配列名で指定される配列は1次元で，SEARCH関数が実行される前に必ずDIM文によって定義されていなければなりません。

・整数値には，探し出す値を整数で指定します。

・開始番号には，配列内のデータを探し始める位置の添字を指定します。

・開始番号の指定を省略すると，配列内データの最初から探し始めます。

・ステップ値には，添字の増加のきざみ値を指定します。

・ステップ値には，負の値および0は指定できません。

・ステップ値の指定を省略すると，「1」が採用されます。

<書き方例>

① X＝SEARCH(A%, 50, 5, 2)

配列A%の中から「50」というデータを，添字を5から2きざみで増加させながら探し，見つかったらそのときの添字を変数Xに求めています。

② Y＝SEARCH(B%, S%)

配列B%内にS%の値と同じ値が含まれているかどうか，添字を配列の先頭から1ずつ増加させて探し，見つかったらそのときの添字を変数Yに求めています。

5.1.4 キーインするプログラム

作成したフローチャートに従って，プログラムをコーディングします。

【 コーディング例 】

```
 10 ' ｳﾘｱｹﾞ ｼｭｳｹｲ ﾌﾟﾛｸﾞﾗﾑ
 20 DIM CD%(10),NM$(10),TN(10)
 30 CLS
 40 LPRINT TAB(7);"***** ｳﾘｱｹﾞ ﾒｲｻｲ *****":LPRINT
 50 ' ﾃﾞｰﾀ
 60 DATA 29,"ﾋﾞﾃﾞｵ",183000
 70 DATA 12,"ｸｰﾗｰ",109800
 80 DATA 16,"ﾄﾞﾗｲﾔｰ",7200
 90 DATA 9,"ﾄｰｽﾀｰ",8800
 100 DATA 3,"ｸﾘｰﾅｰ",16500
 110 DATA 28,"ｽﾃﾚｵ",222000
 120 DATA 10,"ﾃﾚﾋﾞ",79000
 130 DATA 21,"ﾗｼﾞｶｾ",58000
 140 DATA 7,"ﾚｲｿﾞｳｺ",137500
 150 DATA 6,"ﾃﾞﾝｼﾚﾝｼﾞ",123000
 160 ' ﾃﾞｰﾀ ｾｯﾄ
 170 FOR I=1 TO 10
```

（つづく）

```
180   READ CD%(I),NM$(I),TN(I)
190 NEXT I
200 ' ﾆｭｳﾘｮｸ ｼｮﾘ
210 INPUT "ｼｮｳﾋﾝ ｺｰﾄﾞ ";SCD%
220 IF SCD%=999 THEN 420
230 NU=SEARCH(CD%,SCD%)
240 IF NU=-1 THEN PRINT "ｴﾗｰ!!":GOTO 210
250 INPUT "ｽｳﾘｮｳ ";SU:PRINT
260 ' ﾃﾞｰﾀ ﾁｪｯｸ
270 PRINT "ｼｮｳﾋﾝ ｺｰﾄﾞ =";CD%(NU)
280 PRINT "ｼｮｳﾋﾝ ﾒｲ   =";NM$(NU)
290 PRINT "ﾀﾝｶ        =";TN(NU)
300 PRINT "ｽｳﾘｮｳ      =";SU:PRINT
310 PRINT TAB(3);"ｶｸﾆﾝ  Key In [Y] / [N]";
320 INPUT OK$:PRINT:PRINT
330 IF OK$><"Y" AND OK$><"y" THEN 210
340 KIN#=TN(NU)*SU:TOTAL#=TOTAL#+KIN#
350 ' ｼｭﾂﾘｮｸ ｼｮﾘ
360 LPRINT USING "##   ";CD%(NU);
370 LPRINT USING "&         &  ";NM$(NU);
380 LPRINT USING "###,###  ";SU;
390 LPRINT USING "¥¥##,###,###";KIN#
400 GOTO 210
410 ' ｺﾞｳｹｲ ｲﾝｼﾞ
420 LPRINT
430 LPRINT TAB(14);"ｺﾞｳｹｲ";TAB(22);
440 LPRINT USING "¥¥#,###,###,###";TOTAL#
450 END
```

【 コーディング例の説明 】

①準備処理

・プログラムの先頭から行番号190までが，フローチャート例の準備処理に当たります。

・行番号60～150のDATA文で，商品コード，商品名，単価を定義しています。

・行番号170～190では，DATA文で定義した商品コード，商品名，単価をそれぞれ配列CD%，NM$，TNへ読み込んでいます。

②入力処理

・行番号200～340がフローチャート例の入力処理に当たり，行番号210～250がデータ入力の部分で，行番号270～340が入力したデータをチェックする部分です。

・行番号210で商品コードを，行番号230で数量を入力します。

・行番号220では，商品コードに「999」が入力されたときに行番号420へ分岐しています。

・行番号230では，行番号210で入力した商品コードと一致するデータを配列CD%から探しています。

・行番号240では，行番号230で対応するデータが見つからないときに「エラー!!」というメッ

セージを表示し，行番号210へ分岐しています。

・行番号270～300では，確認のために入力したデータと対応するデータを表示しています。

・行番号310～320では，入力したデータが正しいかどうか、確認のコードを入力します。

・行番号330では，行番号320で「Y」または「y」以外のデータが入力されたとき行番号210へ分岐しています。

・行番号340では，売上金額の明細と合計を計算しています。

③出力処理

・行番号350～400が，フローチャート例の出力処理に当たります。

・行番号350～390では，プリンタに明細を印字しています。

・行番号400では，処理を繰り返すために行番号210へ分岐しています。

④終了処理

・行番号410～450が，フローチャート例の終了処理に当たります。

・行番号430～440では，売上合計を印字しています。

設問 44　メモリーをクリアした後，コーディング例のプログラムをキーインして下さい。

・メモリーのクリアは「NEW」コマンドで行います。

設問 45　正しくキーインされているかどうか，プログラムリストをすべてプリンタに印字して下さい。

【 プリンタ印字例 】

```
10 ' ｳﾘｱｹﾞ ｼｭｳｹｲ ﾌﾟﾛｸﾞﾗﾑ
20 DIM CD%(10),NM$(10),TN(10)
30 CLS
40 LPRINT TAB(7);"***** ｳﾘｱｹﾞ ﾒｲｻｲ *****":LPRINT
50 ' ﾃﾞｰﾀ
60 DATA 29,"ﾋﾞﾃﾞｵ",183000
70 DATA 12,"ｸｰﾗｰ",109800
80 DATA 16,"ﾄﾞﾗｲﾔｰ",7200
90 DATA 9,"ﾄｰｽﾀｰ",8800
100 DATA 3,"ｸﾘｰﾅｰ",16500
110 DATA 28,"ｽﾃﾚｵ",222000
120 DATA 10,"ﾃﾚﾋﾞ",79000
130 DATA 21,"ﾗｼﾞｶｾ",58000
140 DATA 7,"ﾚｲｿﾞｳｺ",137500
150 DATA 6,"ﾃﾞﾝｼﾚﾝｼﾞ",123000
```

（つづく）

```
160 ' デ゛ータ セット
170 FOR I=1 TO 10
180   READ CD%(I),NM$(I),TN(I)
190 NEXT I
200 ' ニュウリョク ショリ
210 INPUT "ショウヒン コード゛ ";SCD%
220 IF SCD%=999 THEN 420
230 NU=SEARCH(CD%,SCD%)
240 IF NU=-1 THEN PRINT "エラー!!":GOTO 210
250 INPUT "スウリョウ ";SU:PRINT
260 ' デ゛ータ チェック
270 PRINT "ショウヒン コード゛ =";CD%(NU)
280 PRINT "ショウヒン メイ    = ";NM$(NU)
290 PRINT "タンカ          =";TN(NU)
300 PRINT "スウリョウ        =";SU:PRINT
310 PRINT TAB(3);"カクニン  Key In [Y] / [N]";
320 INPUT OK$:PRINT:PRINT
330 IF OK$><"Y" AND OK$><"y" THEN 210
340 KIN#=TN(NU)*SU:TOTAL#=TOTAL#+KIN#
350 ' シュツリョク ショリ
360 LPRINT USING "##   ";CD%(NU);
370 LPRINT USING "&          &  ";NM$(NU);
380 LPRINT USING "###,###   ";SU;
390 LPRINT USING "¥¥##,###,###";KIN#
400 GOTO 210
410 ' コ゛ウケイ インシ゛
420 LPRINT
430 LPRINT TAB(14);"コ゛ウケイ";TAB(22);
440 LPRINT USING "¥¥##,###,###,###";TOTAL#
450 END
```

5.1.5 作成したプログラムの実行

設 問 46 今メモリーに記憶しているプログラムを，次のデータを使って実行して下さい。

（入力するデータ）

商品コード	数量
12	8
21	15
9	4
29	7
3	12
16	9
10	16
999	

【 実行結果例 】

（画面）

```
ｼｮｳﾋﾝ ｺｰﾄﾞ ? 12
ｽｳﾘｮｳ ? 8

ｼｮｳﾋﾝ ｺｰﾄﾞ = 12
ｼｮｳﾋﾝ ﾒｲ   = ｸｰﾗｰ
ﾀﾝｶ        = 109800
ｽｳﾘｮｳ      = 8

    ｶｸﾆﾝ  Key In [Y] / [N]? Y

ｼｮｳﾋﾝ ｺｰﾄﾞ ? 21
ｽｳﾘｮｳ ? 15

    ｶｸﾆﾝ  Key In [Y] / [N]? y

ｼｮｳﾋﾝ ｺｰﾄﾞ ? 999
Ok
```

（プリンタ）

```
            ***** ｳﾘｱｹﾞ ﾒｲｻｲ *****

 12  ｸｰﾗｰ               8    ¥878,400
 21  ﾗｼﾞｶｾ             15    ¥870,000
  9  ﾄｰｽﾀｰ              4     ¥35,200
 29  ﾋﾞﾃﾞｵ              7  ¥1,281,000
  3  ｸﾘｰﾅｰ             12    ¥198,000
 16  ﾄﾞﾗｲﾔｰ             9     ¥64,800
 10  ﾃﾚﾋﾞ              16  ¥1,264,000

                 ｺﾞｳｹｲ     ¥4,591,400
```

設問 47　正しく実行できていたら，ドライブ２のフロッピーディスクへ「URIAGE.１」という名前でセーブして下さい。

【 実行結果例 】

```
save "URIAGE.1"
Ok
```

5.2　ファンクションキーの利用

現在メモリーに記憶されているプログラムでは，商品コードがわからないと正確にデータを入力することができません。そこで，商品コードの代わりに商品名でも入力できるように修正します。

また，商品名の入力には，キーボード上部のファンクションキーを利用することにします。

5.2.1　プログラム修正の条件

プログラムを修正する条件は，次のとおりです。

【 修正の条件 】

- ファンクションキーにDATA文中の商品名を記憶させます。そのとき，商品名入力時にいちいち⏎キーを押さなくて済むように，リターンコード(⏎キーを押したときに発生するコード)も同時に記憶させます。
- 商品名の入力は，商品コードに「0」が入力されたときのみ実行するようにします。
- 商品名を入力したときは,商品名をキーにして配列から商品コードと単価をサーチします。
- 売上合計を印字した後，ファンクションキーの内容を初期値に戻しておきます。

5.2.2　フローチャートの修正

修正の条件に従って，フローチャートを修正します。修正する部分は，準備処理，入力処理および終了処理です。

(準備処理)

開　始
配列宣言
商品コード、商品名
単価
画面クリア
見出し印字
カウンタを
初期値クリア
配列に
データをセット
プログラム中に
データを定義
ファクションキー
商品名をセット
カウンタに
1を加算
カウンタは
10を越えたか？
NO
YES

(入力処理)

商品コードの入力

商品コードは「999」か？

YES

NO

商品コードは「9」か？

YES

NO

商品名の入力

カウンタ(NU)を初期値クリア

商品名(NU)：入力商品名

＝

≠

カウンタ(NU)に1を加算

カウンタ(NU)は10を超えたか?

NO

YES

NUに－1を代入

商品名，単価を配列からサーチ

配列中に入力した品名コードはあるか?

存在しない

存在する

数量の入力

確認のためのデータ表示

商品コード，商品名
単価，数量

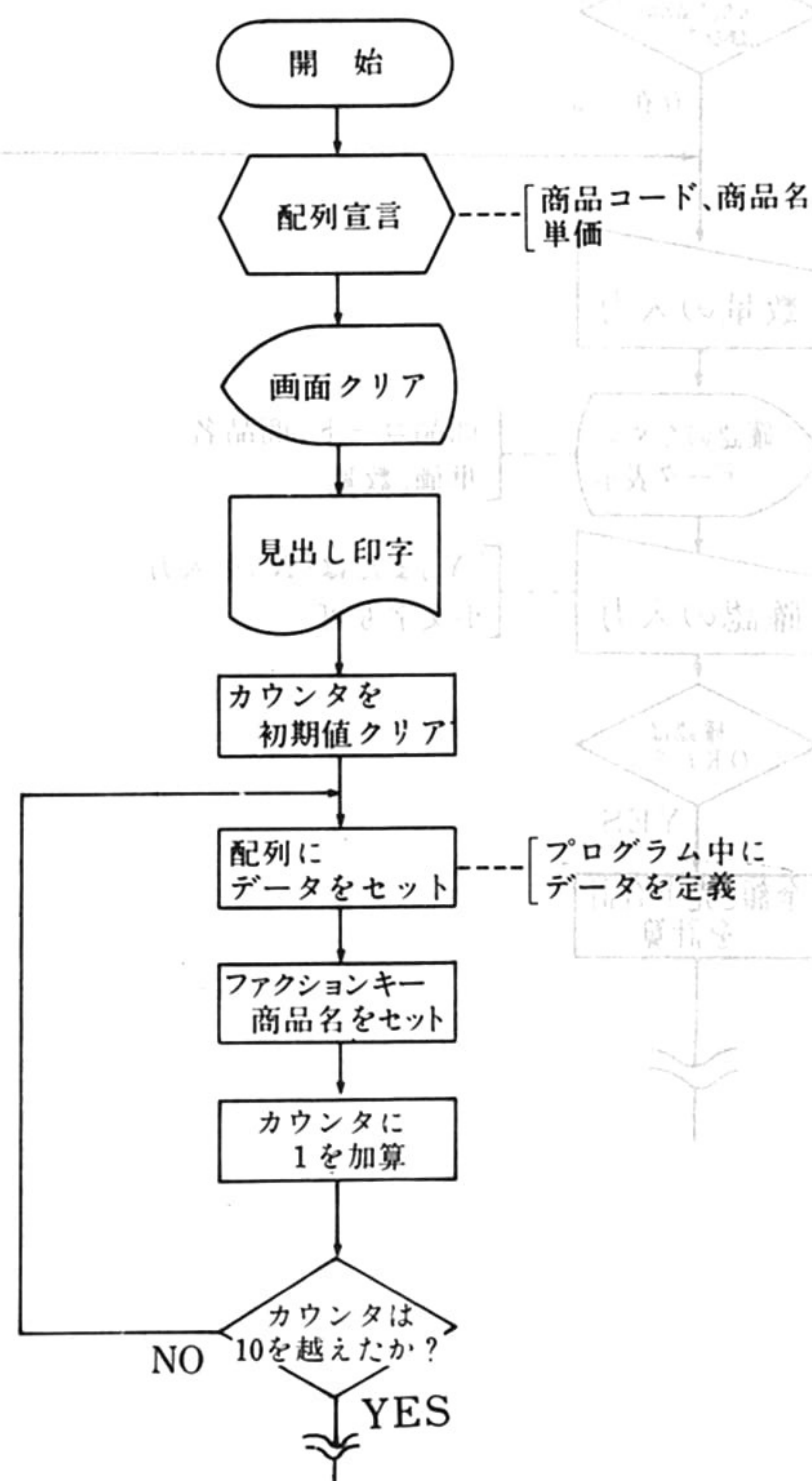
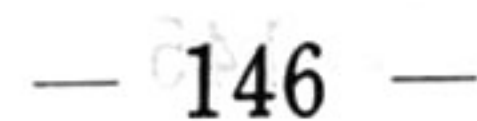
確認の入力

「Y」または「N」を入力
小文字も可

確認はOKか？

NO

YES

金額と売上合計を計算

(終了処理)

5.2.3 プログラムに必要な命令の説明

プログラムを修正するにあたって，新たに必要となる命令や関数を説明します。

(1) ファンクションキーへデータを定義する命令

キーボードの上部にある f・1 ～ f・5 キーは「ファンクションキー」と呼ばれ，ここにはそれぞれ15文字までの文字列を記憶させることができます。なお，ファンクションキーは SHIFT キーと併せて使うことによって f・6 ～ f・10 のキーとして利用することができます。

ファンクションキーにデータを定義するには「KEY」命令を使います。KEY命令の一般形式と書き方例は次のとおりです。

(KEY命令の説明)

<一般形式>

KEY　キー番号，文字列

・ファンクションキーに文字列を定義します。

・キー番号には，1～10の範囲の数値を指定します。

・文字列には，15文字以内の文字列を指定します。

<書き方例>

① KEY 1，"PC-8801mk2"	f・1 キーに「PC-8801mk2」という文字列を定義します。
② KEY N，M＄	変数Nの内容に対応する番号のファンクションキーに，変数M＄の内容を定義します。

ファンクションキーの内容を取り出すには，f・1 ～ f・5 キーを押します。また，SHIFT キーと同時に押すことで f・6 ～ f・10 キーの内容も取り出せます。

そのほか，ファンクションキーの内容をすべて表示する「KEY LIST」命令もあります。KEY LIST命令の一般形式と書き方例は次のとおりです。

(KEY LIST命令の説明)

<一般形式>

KEY LIST

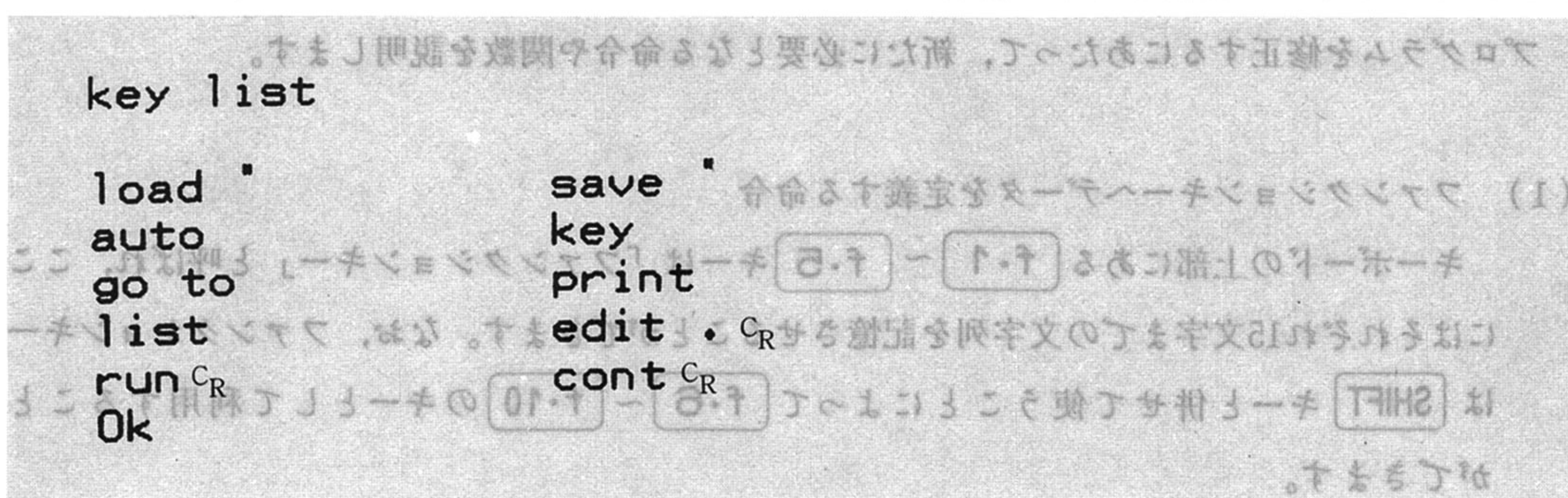
・ファンクションキー(f・1 ～ f・10)の内容をすべて画面に表示します。

<書き方例>

・ KEY LIST

・ファンクションキーが初期値のときKEY LIST命令を実行すると，次のように表示されます。

【 表示例 】

```
key list

load "              save "
auto                key
go to               print
list                edit . CR
run CR              cont CR
Ok
```

【 表示例の説明 】

・左側の五つの文字列が f・1 ～ f・5 キーの内容で，右側の五つの文字列が f・6 ～ f・10 キーの内容です。

・「runC_R」,「contC_R」などの「C_R」とはリターンコードを表していて，⏎ キーを押したときと同じ働きをします。

(2) 文字列の結合

また，命令ではありませんが，PC-8801mkIIには，複数の文字列を一つにつなげる機能もあります。

複数の文字列を一つに結合するには，文字列と文字列を「＋」でつなぎます。文字列の結合の書き方例を次に示します。

(文字列の結合の説明)

<一般形式>

文字列＋文字列＋…

・複数の文字列を一つにつなぎ合わせます。

<書き方例>

① PRINT "PC-"＋"8801"　　画面に「PC-」という文字列と「8801」という文字列をつなげて，「PC-8801」と表示します。

② A＄＝"NEC　" B＄＝"PC-" C＄＝"8801" X＄＝A＄＋B＄＋C＄	変数A＄と変数B＄と変数C＄の内容をつなぎ合わせた結果を変数X＄に代入します。

5.2.4 プログラムの修正

修正したフローチャートに従って、今メモリーに記憶しているプログラムを修正します。

(1)　準備処理部分の修正

[f･1]～[f･10]キーに，DATA文で定義しているデータの内の商品名を記憶させます。そのため，次のように修正して行番号180のREAD命令で配列NM＄に読み込んだ内容を，ファンクションキーへ定義します。その際，入力時に[↵]キーを押さずに済むように，リターンコードも同時に定義します。

```
170 FOR I=1 TO 10
180   READ CD%(I),NM$(I),TN(I)
190 NEXT I
```

⇩

```
170 FOR I=1 TO 10
180   READ CD%(I),NM$(I),TN(I)
185   KEY I,NM$(I)+CHR$(13)
190 NEXT I
```

(2)　入力処理部分の修正

商品コードに「0」が入力されたとき，数量の入力の後に商品名を入力するように，行番号230の後に次の2行を追加します。

```
221 IF SCD%><0 THEN 230
222 INPUT "ｼｮｳﾋﾝ ﾒｲ ";SNM$
```

また，商品名を入力したときは，商品名をキーにしてサーチするように，さらに次の行を追加します。

```
223 FOR NU=1 TO 10
224   IF NM$(NU)=SNM$ THEN 250
225 NEXT NU:NU=-1:GOTO 240
```

行番号235の「NU＝－1」は，一致する商品名が見つからなかったときに行番号250の「エラー!!」というメッセージを表示するためで，「GOTO 250」は，すでに商品名をキーにサーチしているので行番号240のSEARCH関数の実行を避けるためです。

（3） 終了処理部分の修正

売上合計を印字した後，ファンクションキーの内容を初期値に戻すため，行番号450以下に次の処理を追加します。

```
450 ' ﾌｧﾝｸｼｮﾝｷｰ ｵﾌｾｯﾄ
460 KEY 1,"load "+CHR$(34)
470 KEY 2,"auto "
480 KEY 3,"go to "
490 KEY 4,"list "
500 KEY 5,"run"+CHR$(13)
510 KEY 6,"save "+CHR$(34)
520 KEY 7,"key "
530 KEY 8,"print "
540 KEY 9,"edit ."+CHR$(13)
550 KEY 10,"cont"+CHR$(13)
560 END
```

設問 48 今メモリーに記憶されているプログラムを，前の例に従って修正してください。

設問 49 修正が終わったら，正しく修正できているかどうか，プログラムをすべてプリンタに印字して下さい。

【 プリンタ印字例 】

```
10 ' ｳﾘｱｹﾞ ｼｭｳｹｲ ﾌﾟﾛｸﾞﾗﾑ
20 DIM CD%(10),NM$(10),TN(10)
30 CLS
40 LPRINT TAB(7);"***** ｳﾘｱｹﾞ ﾒｲｻｲ *****":LPRINT
50 ' ﾃﾞｰﾀ
60 DATA 29,"ﾋﾞﾃﾞｵ",183000
70 DATA 12,"ｸｰﾗｰ",109800
80 DATA 16,"ﾄﾞﾗｲﾔｰ",7200
90 DATA 9,"ﾄｰｽﾀｰ",8800
100 DATA 3,"ｸﾘｰﾅｰ",16500
110 DATA 28,"ｽﾃﾚｵ",222000
120 DATA 10,"ﾃﾚﾋﾞ",79000
130 DATA 21,"ﾗｼﾞｶｾ",58000
140 DATA 7,"ﾚｲｿﾞｳｺ",137500
150 DATA 6,"ﾃﾞﾝｼﾚﾝｼﾞ",123000
```

（つづく）

```
160 ' ﾃﾞｰﾀ ｾｯﾄ
170 FOR I=1 TO 10
180   READ CD%(I),NM$(I),TN(I)
185   KEY I,NM$(I)+CHR$(13)
190 NEXT I
200 ' ﾆｭｳﾘｮｸ ｼｮﾘ
210 INPUT "ｼｮｳﾋﾝ ｺｰﾄﾞ  ";SCD%
220 IF SCD%=999 THEN 420
221 IF SCD%><0 THEN 230
222 INPUT "ｼｮｳﾋﾝ ﾒｲ  ";SNM$
223 FOR NU=1 TO 10
224   IF NM$(NU)=SNM$ THEN 250
225 NEXT NU:NU=-1:GOTO 240
230 NU=SEARCH(CD%,SCD%)
240 IF NU=-1 THEN PRINT "ｴﾗｰ!!":GOTO 210
250 INPUT "ｽｳﾘｮｳ ";SU:PRINT
260 ' ﾃﾞｰﾀ ﾁｪｯｸ
270 PRINT "ｼｮｳﾋﾝ ｺｰﾄﾞ  =";CD%(NU)
280 PRINT "ｼｮｳﾋﾝ ﾒｲ    = ";NM$(NU)
290 PRINT "ﾀﾝｶ         =";TN(NU)
300 PRINT "ｽｳﾘｮｳ       =";SU:PRINT
310 PRINT TAB(3);"ｶｸﾆﾝ  Key In [Y] / [N]";
320 INPUT OK$:PRINT:PRINT
330 IF OK$><"Y" AND OK$><"y" THEN 210
340 KIN#=TN(NU)*SU:TOTAL#=TOTAL#+KIN#
350 ' ｼｭﾂﾘｮｸ ｼｮﾘ
360 LPRINT USING "##   ";CD%(NU);
370 LPRINT USING "&          &  ";NM$(NU);
380 LPRINT USING "###,###   ";SU;
390 LPRINT USING "¥¥##,###,###";KIN#
400 GOTO 210
410 ' ｺﾞｳｹｲ ｲﾝｼﾞ
420 LPRINT
430 LPRINT TAB(14);"ｺﾞｳｹｲ";TAB(22);
440 LPRINT USING "¥¥#,###,###,###";TOTAL#
450 ' ﾌｧﾝｸｼｮﾝｷｰ ｵﾌｾｯﾄ
460 KEY 1,"load "+CHR$(34)
470 KEY 2,"auto "
480 KEY 3,"go to "
490 KEY 4,"list "
500 KEY 5,"run"+CHR$(13)
510 KEY 6,"save "+CHR$(34)
520 KEY 7,"key "
530 KEY 8,"print "
540 KEY 9,"edit ."+CHR$(13)
550 KEY 10,"cont"+CHR$(13)
560 END
```

【 修正後の説明 】

・行番号185では，行番号180で配列NM＄(I)に読み込んだ商品名を，リターンコードを付けてファンクションキーに定義しています。

・行番号221では，行番号210で商品コードに「0」が入力されていたら行番号222以降の処理を実行し，商品コードが「0」でなければ行番号230へ分岐しています。

・行番号223～225では，行番号222で入力した商品名をキーにしてサーチを行っています。一致する商品名が見つかれば行番号250へ分岐し，見つからなければ行番号240のメッセージを表示します。

・行番号460～550では，ファンクションキーの内容を初期値に戻しています。「CHR＄(13)」はリターンコード，「CHR＄(34)」は「”」の文字を表します。

設問 50　正しく修正できていたら，次のデータを使って実行して下さい。

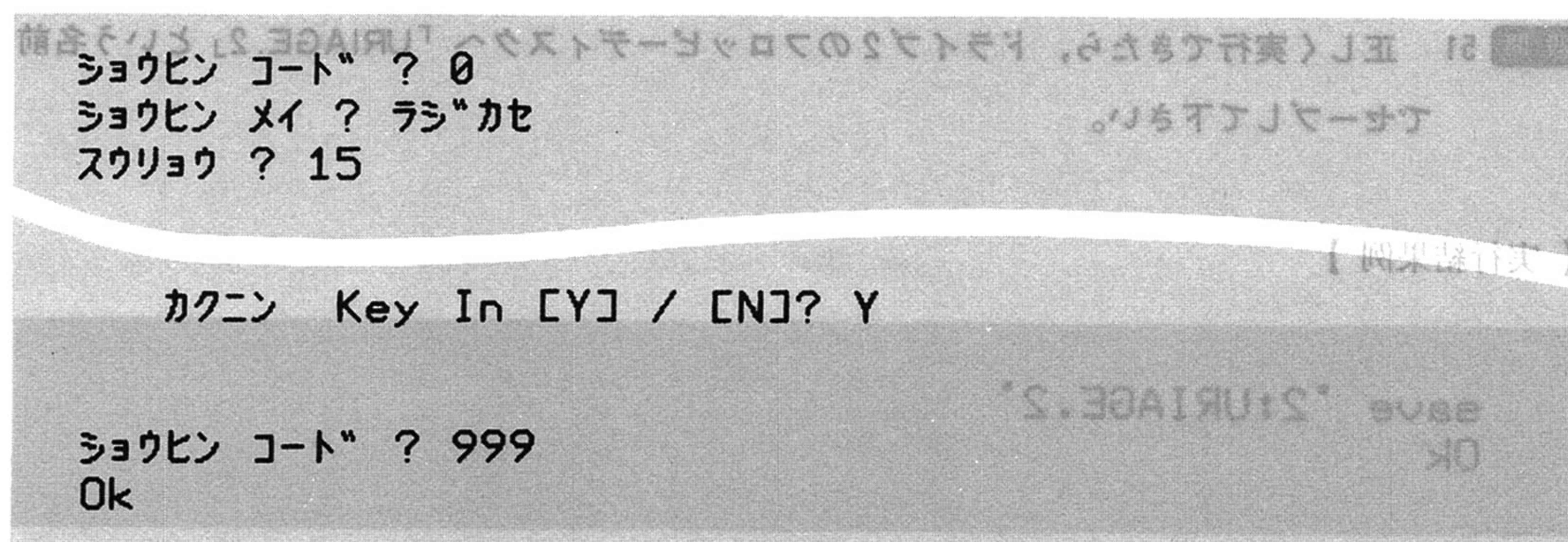
(入力するデータ)

商品コード	商品名	数量
0	f·2	8
0	SHIFT ⊕ f·3	15
9		4
0	f·1	7
3		12
16		9
0	SHIFT ⊕ f·2	16
999		

【 実行結果例 】

(画面)

```
ｼｮｳﾋﾝ ｺｰﾄﾞ ? 0
ｼｮｳﾋﾝ ﾒｲ ? ｸｰﾗｰ
ｽｳﾘｮｳ ? 8

ｼｮｳﾋﾝ ｺｰﾄﾞ  = 12
ｼｮｳﾋﾝ ﾒｲ    = ｸｰﾗｰ
ﾀﾝｶ         = 109800
ｽｳﾘｮｳ       = 8

   ｶｸﾆﾝ  Key In [Y] / [N]? Y
```

(つづく)

```
ｼｮｳﾋﾝ ｺｰﾄﾞ ? 0
ｼｮｳﾋﾝ ﾒｲ ? ﾗｼﾞｶｾ
ｽｳﾘｮｳ ? 15

   ｶｸﾆﾝ  Key In [Y] / [N]? Y

ｼｮｳﾋﾝ ｺｰﾄﾞ ? 999
Ok
```

(プリンタ)

```
             ***** ｳﾘｱｹﾞ  ﾒｲｻｲ  *****

 12   ｸｰﾗｰ                   8       ¥878,400
 21   ﾗｼﾞｶｾ                 15       ¥870,000
  9   ﾄｰｽﾀｰ                  4        ¥35,200
 29   ﾋﾞﾃﾞｵ                  7     ¥1,281,000
  3   ｸﾘｰﾅｰ                 12       ¥198,000
 16   ﾄﾞﾗｲﾔｰ                 9        ¥64,800
 10   ﾃﾚﾋﾞ                  16     ¥1,264,000

                 ｺﾞｳｹｲ             ¥4,591,400
```

【 実行後の説明 】

・商品名の入力は，ファンクションキーを押して行います。ファンクションキーを押すと，そこに定義されている商品名が取り出されます。しかも，その商品名はリターンコードを付けて定義しているため，[⏎] キーを押さずに入力できます。

・ファンクションキーに定義してある商品名は配列NM＄に記憶している商品名と同じなので，商品名の入力にファンクションキーを使えば，入力ミスがなくなります。

・プログラムを実行するとファンクションキーの内容は，次のようになります。

ファンクションキー	内　容
f：1	ビデオC_R
f：2	クーラーC_R
f：3	ドライヤーC_R
f：4	トースターC_R
f：5	クリーナーC_R

ファンクションキー	内　容
f：6	ステレオC_R
f：7	テレビC_R
f：8	ラジカセC_R
f：9	レイゾウコC_R
f：10	デンシレンジC_R

設問 51　正しく実行できたら，ドライブ2のフロッピーディスクへ「URIAGE.2」という名前でセーブして下さい。

【 実行結果例 】

```
save "2:URIAGE.2"
Ok
```

6章　シーケンシャルファイルの利用

今までは，入力したデータを，メモリー内の配列に記憶するか，プリンタに印字するかして処理していました。しかしそれでは，プログラムが終了すると入力したデータを他の目的に利用することができなくなります。そこで，プログラムを保存するのと同じようにデータもフロッピーディスクに保存して，何度も利用できるようにしましょう。

6.1　シーケンシャルファイルの概念

データをフロッピーディスクに保存する方法として，PC-8801mkIIには「シーケンシャルファイル」を作る方法と「ランダムファイル」を作る方法の二通りがあります。ここでは，シーケンシャルファイルについて学習します。ランダムファイルについては，下巻で学習します。

6.1.1　ファイルとは

シーケンシャルファイルやランダムファイルなどの「ファイル」とはどんなものかについて説明します。

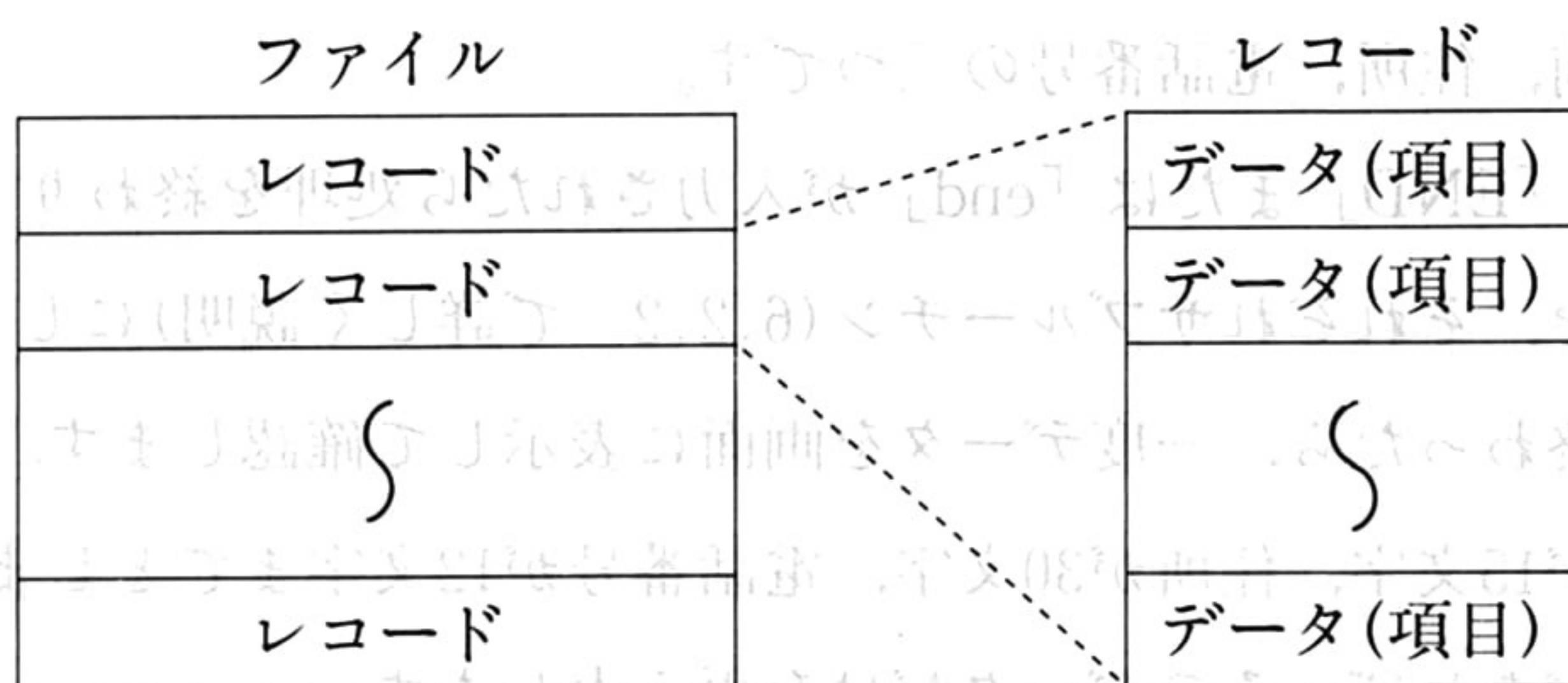
「データ」と呼ばれるものには，さまざまなものがあります。たとえば5章のプログラムを例にとると，「商品コード」，「商品名」，「単価」，「数量」のそれぞれが一つ一つの「データ」ということになります。

また，そのデータが集まり一件分の情報を表す一つのグループができますが，そのグループを「レコード」と呼びます。

さらに，そのレコードがいくつか集まり，一つの「ファイル」が形成されるわけです。

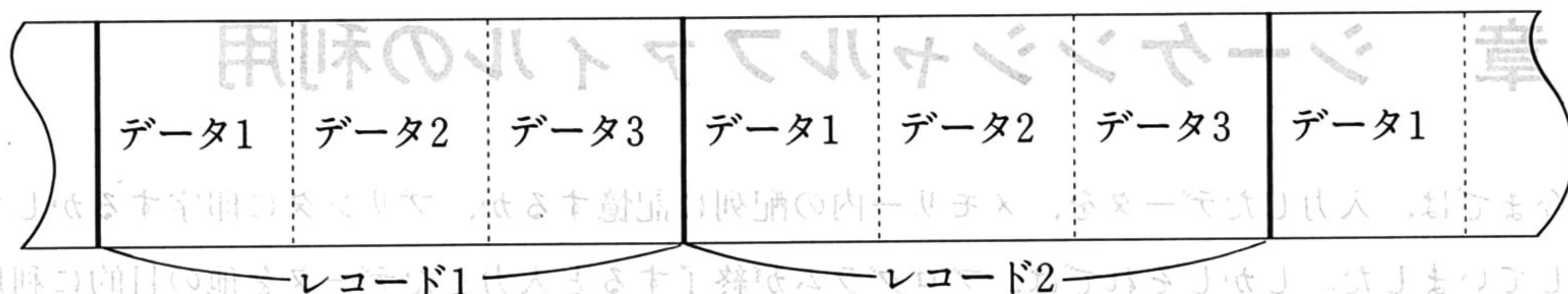
ファイルとレコードとデータの関係は，次のようなイメージになります。

ファイル
レコード
レコード
∫
レコード

レコード
データ(項目)
データ(項目)
∫
データ(項目)

6.1.2　シーケンシャルファイル

シーケンシャルファイルとは，音楽を録音するカセットテープのように，いくつかのデータを発生した順番にフロッピーディスクへ書くことによってできたファイルです。

シーケンシャルファイルは，次のようなイメージで作られます。

データ1	データ2	データ3	データ1	データ2	データ3	データ1
レコード1			レコード2			

また，このようにして作られたシーケンシャルファイルのデータは，書かれた順序でしか読むことができません。書かれたときの順序に逆らって順不同に読んだり，途中のデータから読み始めたりすることはできません。

6.2 プログラム例(5)――住所録プログラム

シーケンシャルファイルの学習として、簡単な住所録プログラムを作成してみましょう。

6.2.1 プログラムに必要な条件

住所録を作るには，データをファイルへ記録(登録)するものと，記録したファイルからデータを読んでプリンタへ出力するものと，後からファイルへデータを追加するものとの三つのプログラムが必要になります。そこで，住所録プログラムを作成するために必要なそれぞれの条件を述べると次のようになります。

【 作成の条件 】

(1) ファイル作成プログラム

・キーボードからデータを入力し，フロッピーディスクへシーケンシャルファイルとして出力します。

・シーケンシャルファイルは，ドライブ2のフロッピーディスクに「JU-F」という名前を付けて作ります。

・入力するデータは，名前，住所，電話番号の三つです。

・名前を入力するときに，「END」または「end」が入力されたら処理を終わります。

・データの入力処理部分を，それぞれサブルーチン(6.2.2 で詳しく説明)にします。

・三つのデータの入力が終わったら，一度データを画面に表示して確認します。

・データの表示は，名前が15文字，住所が30文字，電話番号が12文字までとします。

・入力したデータに誤りがあれば，そのデータだけを再入力します。

・入力したデータに誤りがなければ，確認の入力で ⏎ キーだけを押して，シーケンシャルファイルに書き込みます。

・シーケンシャルファイルには，次のようにデータが書かれます。

名前	住所	電話番号	名前	住所	電話番号	名前
1件目			2件目			

(2) ファイル入力プログラム

・ドライブ2のフロッピーディスクの「JU-F」というシーケンシャルファイルから，データを読み込み，プリンタに印字します。

・プリンタに印字する形式は次のとおりです。

****** ジュウショロク *****
ナマエ ジュウショ Tel－No
17 49

(3) データ追加プログラム

・ドライブ2のフロッピーディスクの「JU-F」というシーケンシャルファイルにデータを追加できるようにします。

・その他の条件は「ファイル作成プログラム」と同様です。

6.2.2 プログラムに必要な命令の説明

住所録を作成するプログラムと印字のプログラムおよびデータを追加するプログラムのそれぞれに必要な命令について，以下に説明します。

(1) ファイルのオープンとクローズ

フロッピーディスクのファイルへデータを書いたり読んだりする場合，たとえば本を読むときに必ず始めに表紙を開き，読み終わると最後に表紙を閉じますが，この動作と全く同じように，ファイルのオープン(OPEN)およびクローズ(CLOSE)を必ず行わなければなりません。また，ファイルをオープンするときは，データを書くためにオープンするのか，既に書かれているデータを読むためにオープンするのか，あるいは既に書かれているデータを消さないでその後へ追加してデータを書くためにオープンするのか等を区別しなければなりません。

シーケンシャルファイルをオープンする「OPEN」命令の一般形式と書き方例を次に示します。

(OPEN命令の説明)

<一般形式>

OPEN ファイルディスクリプタ FOR モード AS #ファイル番号

・シーケンシャルファイルを目的に合ったモードでオープンし，そのファイルに番号を付けます。

・ファイルディスクリプタには，フロッピーディスクのドライブ番号やファイルの名前等を指定します。

・モードは，シーケンシャルファイルの扱い方を決めるもので，次の3種類のうちいずれか一つを指定します。

INPUT　シーケンシャルファイルからデータを読むために，ファイルをオープンする。

OUTPUT　シーケンシャルファイルへデータを書くために，ファイルをオープンする。

APPEND　既にデータが書かれているシーケンシャルファイルの後へデータを追加するために，ファイルをオープンする。

・ファイル番号は，プログラムの中で複数のファイルを使用するときに，他のファイルと区別できるような番号を指定します。

・ファイル番号には，「1」から「How many files(1-15)？」で指定した値までが指定できます。「How many ～」で ⏎ キーだけを押したときは，1～2の範囲になります。

<書き方例>

① OPEN "2：KYUYO" FOR OUTPUT AS #1

ドライブ2のフロッピーディスクに「KYUYO」という名前のファイルをOUTPUTモードでオープンしています。また，ファイル番号には「1」を指定しています。

② OPEN "2：ZAIKO" FOR INPUT AS #1

ドライブ2のフロッピーディスクの「ZAIKO」という名前のファイルをINPUTモードでオープンしています。また，ファイル番号には「1」を指定しています。

③ OPEN "2：JINJI" FOR APPEND AS #2

ドライブ2のフロッピーディスクの「JINJI」という名前のファイルをAPPENDモードでオープンしています。また，ファイル番号には「2」を指定しています。

また，オープンされているファイルをクローズする「CLOSE」命令の一般形式と書き方例は次のとおりです。

(CLOSE命令の説明)

<一般形式>

CLOSE#ファイル番号

- 現在オープンされているファイルをクローズします。
- ファイル番号には，クローズしたいファイルに対して，OPEN命令で指定したファイル番号と同じ番号を指定します。
- ファイル番号を省略したときは，プログラムの中でオープンされているすべてのファイルをクローズします。

<書き方例>

① CLOSE #1　　ファイル番号「1」のファイルをクローズします。

② CLOSE #2　　ファイル番号「2」のファイルをクローズします。

③ CLOSE　　現在オープンされているすべてのファイルをクローズします。

(2) シーケンシャルファイルへデータを出力する命令

シーケンシャルファイルへデータを出力する命令には，「WRITE#」命令と「PRINT#」命令の2種類があります。機能には大差はありませんが，ここでは書式が簡単な方の「WRITE#」命令について説明します。

WRITE#命令の一般形式と書き方例は，次のとおりです。

(WRITE#命令の説明)

<一般形式>

WRITE#ファイル番号，式の列

- シーケンシャルファイルへ，式の列で指定した内容(データ)を出力します。
- ファイル番号には，OPEN命令で指定したファイル番号と同じものを指定します。
- 式の列には，出力するデータが記憶されている変数の名前を，出力する順序に従って記述します。
- 式の列に記述する変数名は，「,」または「;」で区切ります。

<書き方例>

① WRITE#1，A%，B，C$　　ファイル番号「1」のシーケンシャルファイルへ変数A%，B，C$の各々の内容をこの順序で出力します。

② WRITE#2,CD;NM$;GAKU#　　ファイル番号「2」のシーケンシャルファイルへ変数CD,NM$,GAKU#の内容をこの順序で出力します。

（3） シーケンシャルファイルからデータを入力する命令

シーケンシャルファイルからデータをメモリーへ読み込むには，「INPUT#」命令を使います。INPUT#命令の一般形式と書き方例は，次のとおりです。

(INPUT#命令の説明)

＜一般形式＞

INPUT#ファイル番号，式の列

- 指定したファイル番号のファイルを読んで，式の列で示される変数へデータを移します。
- ファイル番号には，オープン命令で指定した番号と同じものを指定します。
- 式の列には，シーケンシャルファイルから読んだデータを記憶するための変数を指定します。この場合の変数の型はデータをファイルへ記録したときと同じでなければなりません。
- 式の列に記述する変数名は，「，」で区切ります。

＜書き方例＞

① INPUT#1，A%，B＄，C，D#	ファイル番号「1」のシーケンシャルファイルからデータを読んで，変数A%，B＄，C，D#へ記憶させます。
② INPUT＃2，X1，X2%，YZ＄	ファイル番号「2」のシーケンシャルファイルからデータを読んで，変数X1，X2%，YZ＄へ記憶させます。

（4） シーケンシャルファイルの終了を判断する関数

シーケンシャルファイルを読んでいてファイルが終わりに達したとき，それを判断する機能が必要になります。PC-8801mkⅡには，ファイルの終了を判断するための機能として「EOF」関数があります。

EOF関数の一般形式と書き方例は，次のとおりです。

(EOF関数の説明)

＜一般形式＞

EOF(ファイル番号)

- ファイル番号で指定したシーケンシャルファイルが終了したときに，終了コードを与えます。

＜書き方例＞

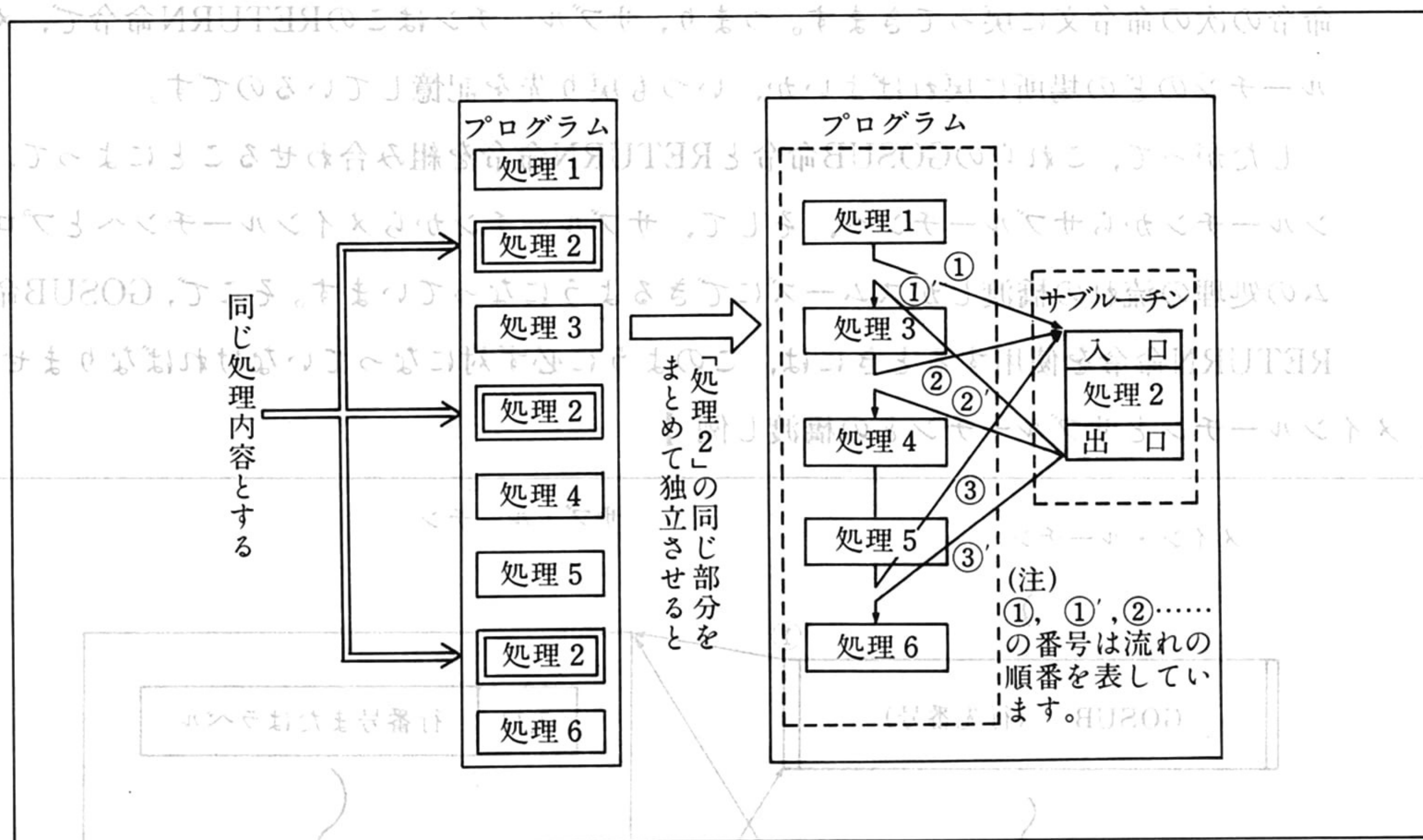
- IF EOF(1) THEN CLOSE #1

  ファイル番号1のシーケンシャルファイルが終了しているならば，そのファイルをクローズします。

(5) サブルーチン

①サブルーチンの考え方

サブルーチンの考え方とその効果などについて，次の図を見ながら説明することにします。

プログラム
処理 1
処理 2
処理 3
処理 2
処理 4
処理 5
処理 2
処理 6
同じ処理内容とする
「処理2」の同じ部分をまとめて独立させると
プログラム
処理 1
処理 3
処理 4
処理 5
処理 6
①
①′
②
②′
③
③′
サブルーチン
入　口
処理 2
出　口
(注)
①，①′，②……の番号は流れの順番を表しています。

サブルーチンの概念図

図の左の概念的なプログラムの中にある処理を分類すると，「処理1」，「処理2」，「処理3」，「処理4」，「処理5」，「処理6」と全部で六つの処理があります。そして，この中には「処理2」のように同じ処理ができる命令文の集まりが3か所もあります。そこで，この同じ処理ができる処理2の命令文を概念図の右のようにまとめて，これを必要なときに利用するようにします。

そうすると，プログラム全体の大きさは重複部分がなくなり，その分だけプログラムが小さくて済みます。そして，その結果プログラミング作業もたいへん楽になると同時に，メモリーの節約にもなってきます。

このように，あるプログラムの中から処理が共通している部分を取り出して独立させたものを「サブルーチン」と呼んでいます。

また，サブルーチンに対して，共通の処理を含まない残った処理の集まりを「メインルーチン」と呼んでいます。

②メインルーチンとサブルーチンのやりとり

次に，メインルーチンとサブルーチンの橋渡しはどのようにして行うか説明します。それには，下の図のようにプログラムの中で「GOSUB」命令と「RETURN」命令を使用し

ます。メインルーチンからサブルーチンを利用するには，GOSUB命令を使います。そして，この命令のあとに利用しようとするサブルーチンの入口を示す開始行番号を書きます。

次に，サブルーチン内での処理が終わってメインルーチンに流れを戻すには，サブルーチンの最後にRETURN命令を書きます。すると，流れはサブルーチンを利用したGOSUB命令の次の命令文に戻ってきます。つまり，サブルーチンはこのRETURN命令で，メインルーチンのどの場所に戻ればよいか，いつも戻り先を記憶しているのです。

したがって，これらのGOSUB命令とRETURN命令を組み合わせることによって，メインルーチンからサブルーチンへ，そして，サブルーチンからメインルーチンへとプログラムの処理の流れの橋渡しがスムーズにできるようになっています。そこで，GOSUB命令とRETURN命令を使用するときには，このように必ず対になっていなければなりません。

【 メインルーチンとサブルーチンとの橋渡し例 】

メイン・ルーチン

GOSUB　(行先番号)　①

②

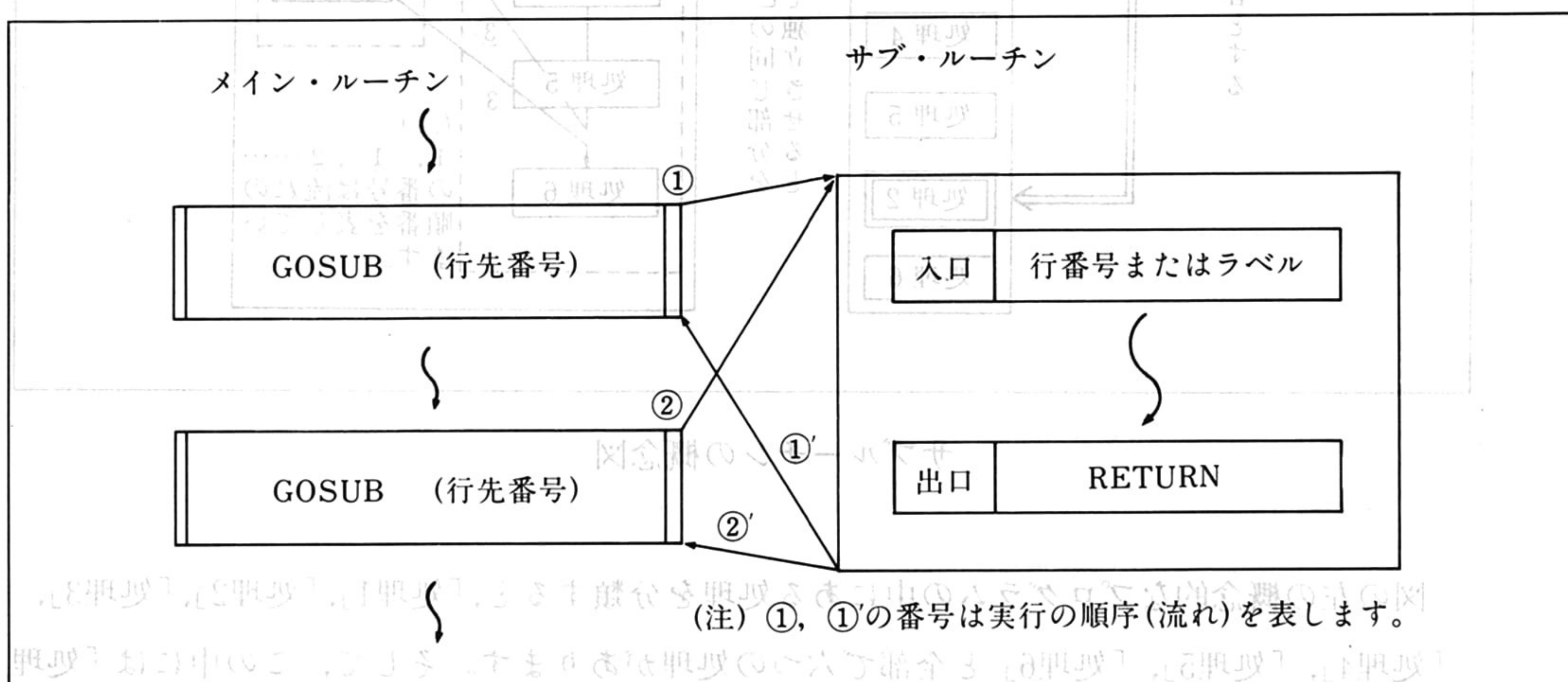
GOSUB　(行先番号)　②′

サブ・ルーチン

入口　行番号またはラベル

①′

出口　RETURN

(注) ①，①′の番号は実行の順序(流れ)を表します。

GOSUB命令とRETURN命令の一般形式と書き方例を次に示します。

(GOSUB命令とRETURN命令の説明)

<一般形式1>

GOSUB　サブルーチン開始行番号

・メインルーチンから指定されたサブルーチン開始行番号のサブルーチンへ分岐します。

・サブルーチン開始行番号の代わりにラベル名も使用できます。

<一般形式2>

RETURN　行番号

・サブルーチンからメインルーチンへ戻ります。

・この命令は，GOSUB命令と正しく対応していなければなりません。

・RETURN命令を実行すると，対応するGOSUB命令の次の命令を実行します。

・通常は行番号は省略します。

・行番号を付けると，指定された行へ強制的に戻ります。

・行番号の代わりにラベル名も使用できます。

<書き方例>

① 50 GOSUB 200 ― 行番号50で行番号200へ分岐し，行番号200～220のサブルーチンを実行した後GOSUB命令の次に書かれている命令文の実行に戻ります。

```
50   GOSUB 200
      ⁓
200  FOR J=2 TO MR STEP 2
210  PRINT "■";:NEXT J
220  RETURN
```

② 100 GOSUB ＊SUB.RTN ― 行番号100でラベル名＊SUB. RTNの行へ分岐し，行番号500～550のサブルーチンを実行した後GOSUB命令の次に書かれている命令文の実行に戻ります。

```
100  GOSUB ＊SUB.RTN
      ⁓
500  ＊SUB.RTN
      ⋮
550  RETURN
```

(6) 複数のサブルーチンを指定する命令

今まで学習してきた，GOTO命令，IF命令およびこれから学習するGOSUB命令は，いずれもプログラムの流れを変える命令です。

このほかにも，プログラムの流れを変える命令として，「ON…GOTO」命令，「ON…GOSUB」命令があります。

このうち，今回のプログラムでは主にGOSUB命令とON…GOSUB命令を使用します。ON…GOSUB命令は，複数個のサブルーチンを用意しておき，条件によってそのいずれかのサブルーチンに分岐するものです。

ON…GOSUB命令の一般形式と書き方例は次のとおりです。

(ON…GOSUB命令の説明)

<一般形式>

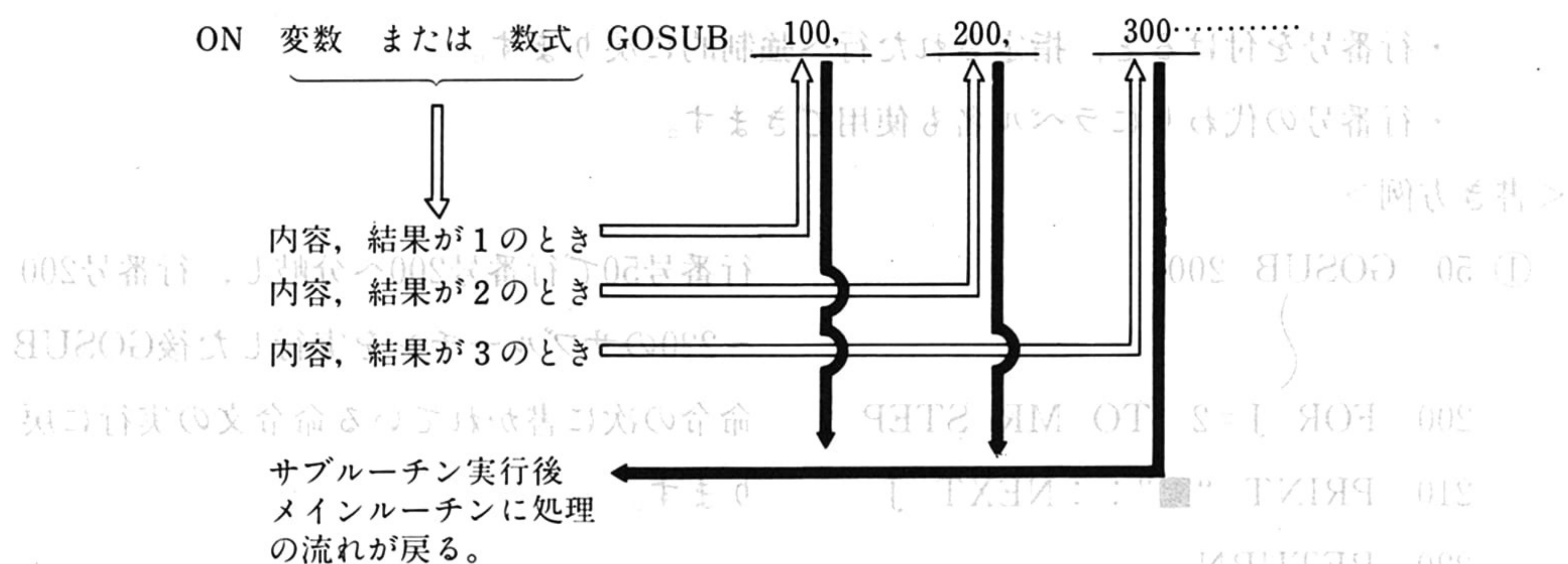
ON 式 GOSUB 行番号$_1$，行番号$_2$，行番号$_3$，……

・式の値に対応する並びの行番号で始まるサブルーチンへ一時分岐して，「RETURN」命令で戻ります。

・式の値が「1」のときは行番号$_1$のサブルーチンへ，「2」のときは行番号$_2$のサブルーチンへ分岐します。

ON　変数　または　数式　GOSUB　100,　200,　300…………

内容，結果が１のとき
内容，結果が２のとき
内容，結果が３のとき

サブルーチン実行後
メインルーチンに処理
の流れが戻る。

・式には，0～255の範囲の値を指定します。
・行番号の代わりにラベル名も使用できます。
・一つのON…GOSUB命令中には，行番号とラベル名を混在して指定することはできません。

<書き方例>

① ON N GOSUB 100, 200, 300

Nの値が「1」のときは行番号100のサブルーチンを，「2」のときは行番号200のサブルーチンを，「3」のときは行番号300のサブルーチンをそれぞれ実行します。

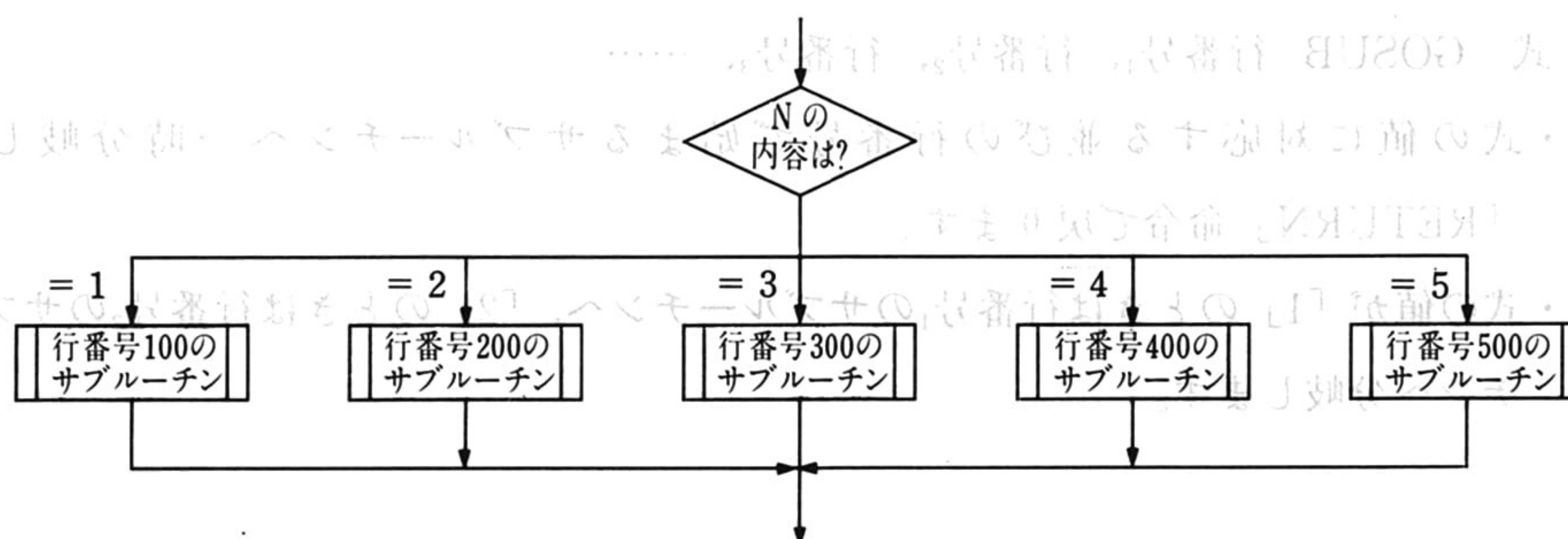
② ON A+B GOSUB ＊LABEL1, ＊LABEL2, ＊LABEL3

A+Bの値が「1」のときはラベル名＊LABEL1のサブルーチンを，「2」のときはラベル名＊LABEL2のサブルーチンを，「3」のときはラベル名＊LABEL3のサブルーチンをそれぞれ実行します。

また，ON…GOSUB命令は，フローチャートでは次のように表します。

<フローチャート例>

・ON N GOSUB 100, 200, 300, 400, 500 の場合

Nの
内容は?
＝1　＝2　＝3　＝4　＝5
行番号100の サブルーチン
行番号200の サブルーチン
行番号300の サブルーチン
行番号400の サブルーチン
行番号500の サブルーチン

(7) 複数個の分岐先を指定する命令

また，ON…GOSUB命令によく似た命令に，「ON…GOTO」命令があります。

ON…GOTO命令は，複数個の分岐先を用意しておき，条件によってそのいずれかの分岐先へ分岐します。

ON…GOTO命令の一般形式と書き方例は次のとおりです。

なおこの命令は，今回のプログラムには使用しません。

(ON…GOTO命令の説明)

＜一般形式＞

ON 式 GOTO 行番号$_1$，行番号$_2$，行番号$_3$，……

・式の値に対応する並びの行番号に分岐します。

・式の値が「1」のときは行番号$_1$へ、「2」のときは行番号$_2$へ分岐します。

ON 変数または数式 GOTO 100， 200， 300

内容・結果が1の時 → 100

内容・結果が2の時 → 200

内容・結果が3の時 → 300

・式には，0～255の範囲の値を指定します。

・行番号の代わりにラベル名を指定しても構いません。

・一つの命令文中には，行番号とラベル名を混在して指定することはできません。

＜書き方例＞

① ON N GOTO 100，200，300

Nの値が「1」のときは行番号100へ，「2」のときは行番号200へ，「3」のときは行番号300へ分岐します。

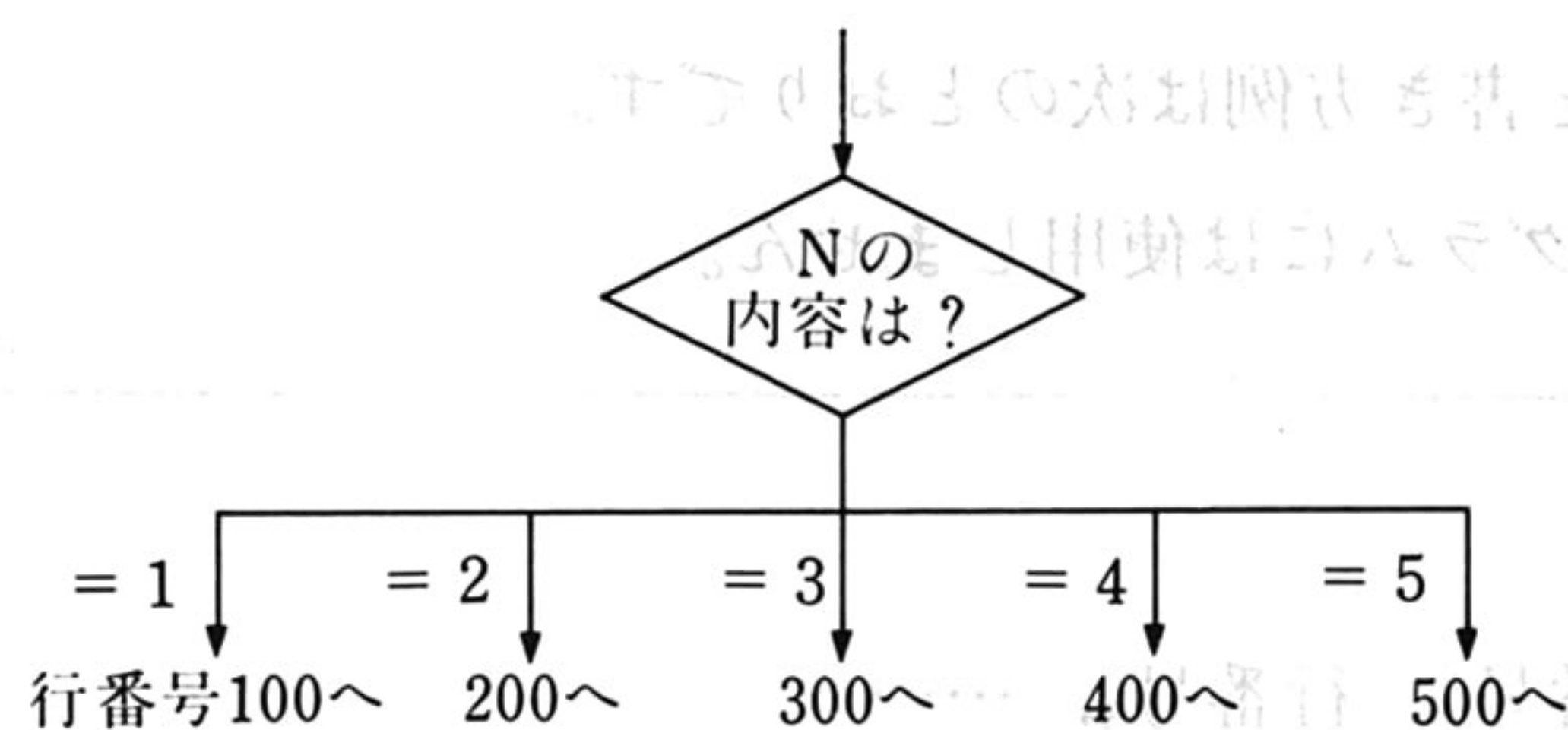
② ON A+B GOTO ＊LABEL1，＊LABEL2，＊LABEL3

A+Bの値が「1」のときはラベル名＊LABEL1の行へ，「2」のときはラベル名＊LABEL2の行へ，「3」のときはラベル名＊LABEL3の行へ分岐します。

また，ON…GOTO命令は，フローチャートでは次のように表します。

<フローチャート例>

・ON N GOTO 100, 200, 300, 400, 500 の場合

```
            │
            ▼
       ／ Nの   ＼
      ＜ 内容は？ ＞
       ＼       ／
            │
  ┌─────┬─────┼─────┬─────┐
 =1    =2    =3    =4    =5
  ▼     ▼     ▼     ▼     ▼
行番号100へ 200へ 300へ 400へ 500へ
```

（8） 空白(スペース)文字を確保する関数

連続した幾つかのスペースを作るには「SPACE＄」関数を使います。

SPACE＄関数の一般形式と書き方例は次のようになります。

(SPACE＄関数の説明)

<一般形式>

SPACE＄(数式)

・数式の値で表される長さの空白文字列を与えます。

・数式には，0～255の範囲の値を指定します。

<書き方例>

① PRINT SPACE＄(20) ； “ABC”

画面に20文字の空白文字列に続いて「ABC」の文字を表示します。

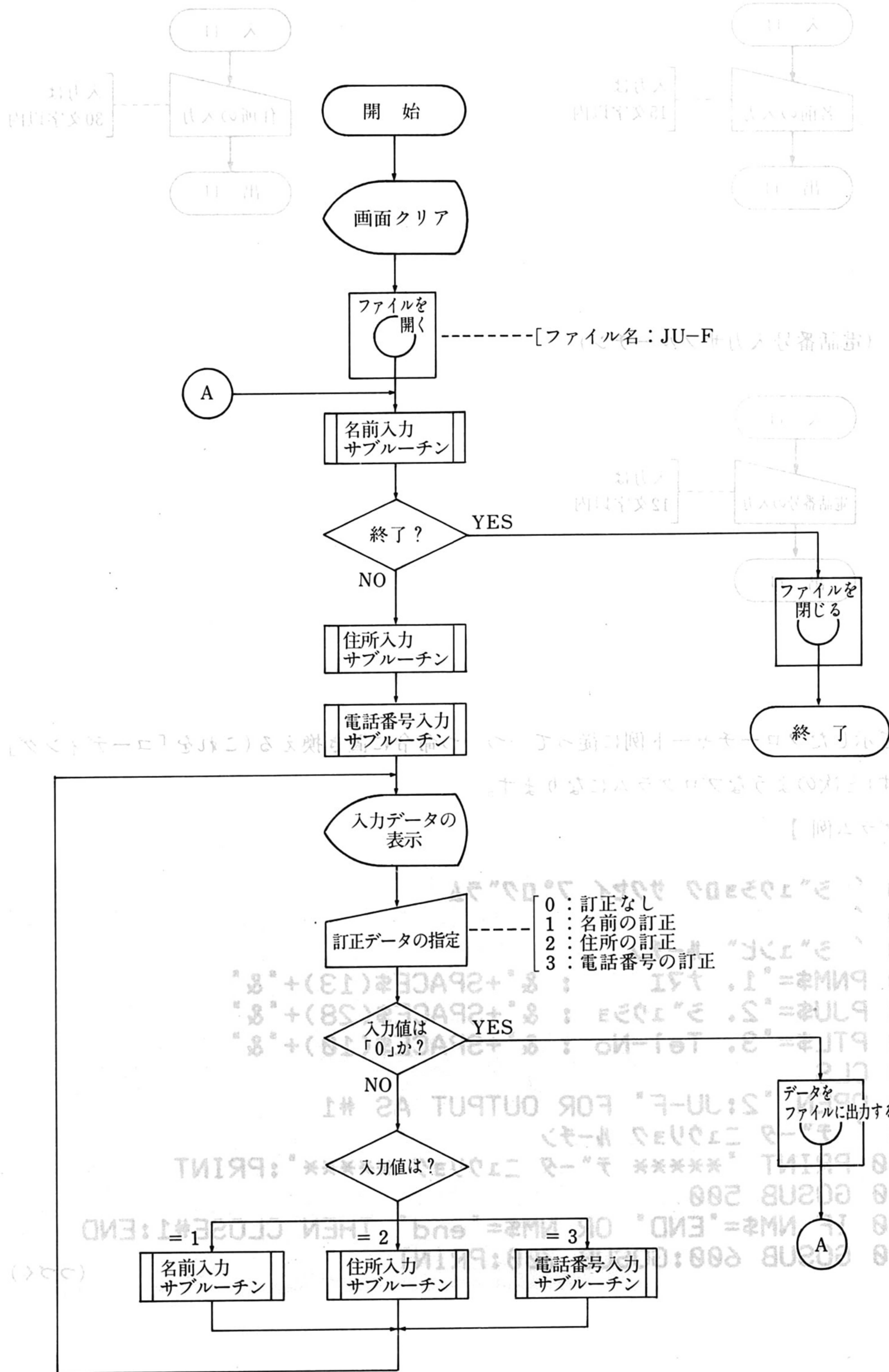
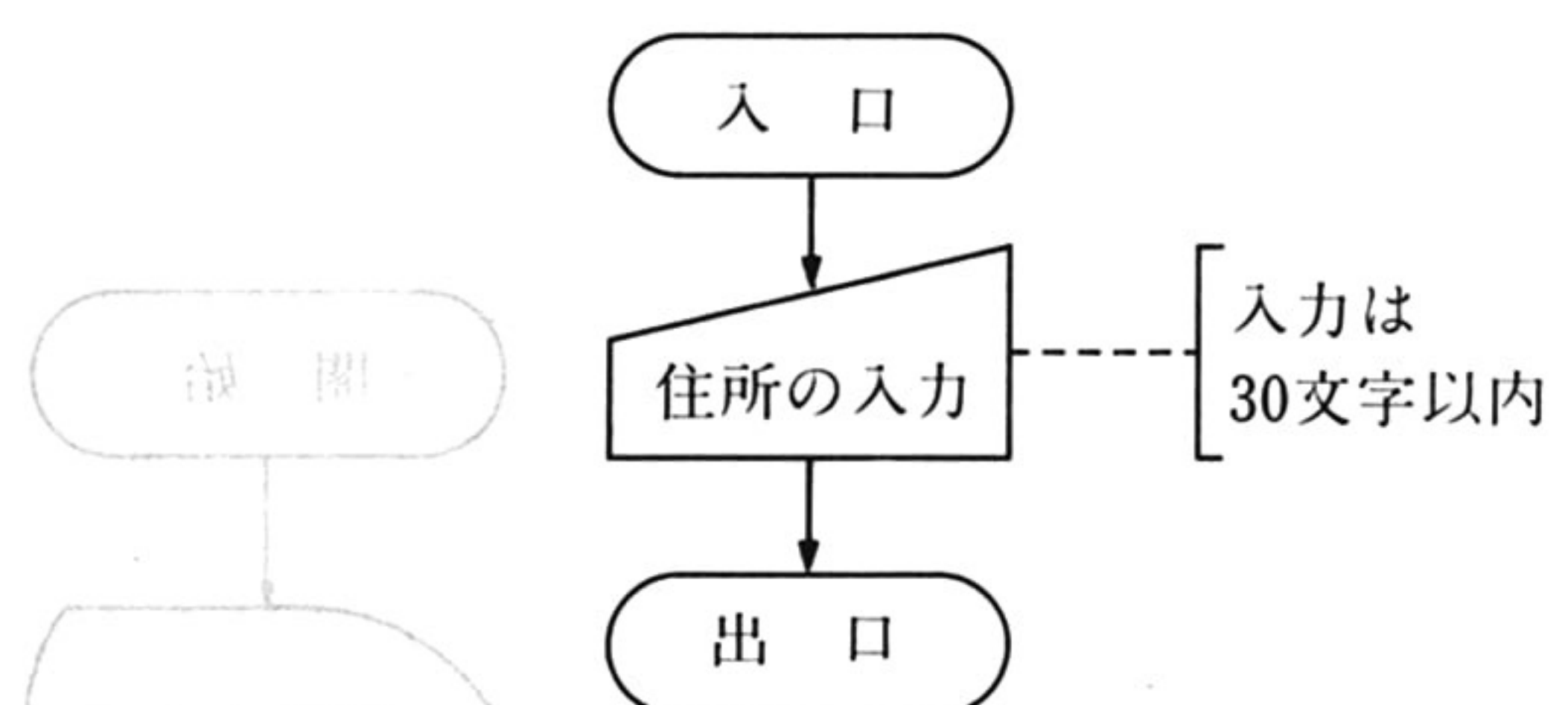
② A＄＝SPACE＄(30)

変数A＄に30文字の空白文字列を代入します。

6.2.3 シーケンシャルファイルの作り方

フロッピーディスクにシーケンシャルファイルを作成するには，「ファイル作成プログラム」が必要になります。そこで，6.2.1の(1)で述べた「ファイル作成プログラム」の作成条件を満たすプログラムのフローチャート例を次に示します。

【 フローチャート例 】

・主(メイン)ルーチン

開　始

画面クリア

ファイルを開く　[ファイル名：JU-F

A

名前入力サブルーチン

終了？　YES　→　ファイルを閉じる　→　終　了

NO

住所入力サブルーチン

電話番号入力サブルーチン

入力データの表示

訂正データの指定

[0 ：訂正なし
1 ：名前の訂正
2 ：住所の訂正
3 ：電話番号の訂正

入力値は「0」か？　YES　→　データをファイルに出力する　→　A

NO

入力値は？

= 1　名前入力サブルーチン

= 2　住所入力サブルーチン

= 3　電話番号入力サブルーチン

・副(サブ)ルーチン

(名前入力サブルーチン)

入　口

名前の入力 ---- 入力は 15文字以内

出　口

(住所入力サブルーチン)

入　口

住所の入力 ---- 入力は 30文字以内

出　口

(電話番号入力サブルーチン)

入　口

電話番号の入力 ---- 入力は 12文字以内

出　口

ここで示したフローチャート例に従って一つ一つ命令に置き換える(これを「コーディング」と呼びます)と次のようなプログラムになります。

【 プログラム例 】

```
10 ' ジュウショロク サクセイ プログラム
20 '
30 ' ジュンビ ルーチン
40 PNM$="1. ナマエ     : &"+SPACE$(13)+"&"
50 PJU$="2. ジュウショ : &"+SPACE$(28)+"&"
60 PTL$="3. Tel-No : &"+SPACE$(10)+"&"
70 CLS
80 OPEN "2:JU-F" FOR OUTPUT AS #1
90 ' データ ニュウリョク ルーチン
100 PRINT "***** データ ニュウリョク *****":PRINT
110 GOSUB 500
120 IF NM$="END" OR NM$="end" THEN CLOSE#1:END
130 GOSUB 600:GOSUB 700:PRINT
```

(つづく)

```
140 ' ﾃﾞｰﾀ ﾁｪｯｸ ﾙｰﾁﾝ
150 PRINT USING PNM$;NM$
160 PRINT USING PJU$;JUSHO$
170 PRINT USING PTL$;TEL$:PRINT
180 PRINT TAB(3);"ﾃｲｾｲ ｱﾘ [1-3]"
190 PRINT TAB(8);"ﾅｼ [RET] ";:INPUT C:PRINT
200 IF C<0 OR C>3 THEN PRINT "ｴﾗｰ!!":GOTO 180
210 IF C=0 THEN 240
220 ON C GOSUB 500,600,700:PRINT
230 GOTO 150
240 WRITE#1,NM$,JUSHO$,TEL$:PRINT
250 GOTO 100
500 ' ﾅﾏｴ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
510 INPUT "ﾅﾏｴ = ";NM$
520 RETURN
600 ' ｼﾞｭｳｼｮ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
610 INPUT "ｼﾞｭｳｼｮ = ";JUSHO$
620 RETURN
700 ' Tel-No ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
710 INPUT "Tel-No = ";TEL$
720 RETURN
```

【 プログラム例の説明 】

・キーボードから名前，住所，電話番号を入力し，間違っていないかチェックした後，フロッピーディスクのシーケンシャルファイルへ出力します。

・行番号30～80が，準備処理部分に当たります。

・行番号40～60では，入力データをチェックするときの表示に使う編集文字列を変数PNM＄，PJU＄，PTL＄に定義しています。編集文字列はそれぞれ次のようになります。

PNM＄――”1．ナマエ　：&　　　　&”（15文字）

PJU＄――”2．ジュウショ：&　　　　　　　&”（30文字）

PTL＄――”3．Tel-No　：&　　　&”（12文字）

・行番号80では，ドライブ2のフロッピーディスクに「JU-F」という名前のシーケンシャルファイルをOUTPUTモードでファイル番号「1」にオープンしています。

・行番号90～130が，入力処理部分に当たります。

・行番号110の「GOSUB 500」と，行番号130の「GOSUB 600」と「GOSUB 700」では，それぞれ名前，住所電話番号を入力するサブルーチンを実行しています。

・行番号120では，名前の入力時に「END」または「end」が入力されたら，ファイル番号「1」のファイルを閉じて処理を終了するようにしています。

・行番号140～230が，データチェック処理部分に当たります。

・行番号150～170では，確認用に今入力したデータを編集して表示しています。編集文字列は，行番号40～60で変数のPNM＄，PJU＄，PTL＄に定義したものを使っています。

・行番号190の「INPUT　C」では，「1」～「3」の数値を入力するか，↵キーを押します。

・INPUT命令の実行で↵キーだけを押すと，数値変数には「0」が，文字変数には「ヌルキャラクタ」が入ります。

・行番号200では，行番号190で変数Cに0～3の範囲外のデータが入力されたときにエラーと判断し，確認用の入力を再実行しています。

・行番号210は，行番号190の「INPUT　C」で↵キーだけが押されたときの判断で，変数Cに「0」が入っていればシーケンシャルファイルへデータを出力します。

・行番号240～250が，ファイル出力処理部分です。

・行番号240では，入力したデータを名前，住所，電話番号の順にファイル番号1のシーケンシャルファイルへ出力しています。

・行番号500～520が，名前を入力するサブルーチンです。

・行番号600～620が，住所を入力するサブルーチンです。

・行番号700～720が，電話番号を入力するサブルーチンです。

設問 52　**メモリーをクリアした後，コーディングしたプログラムをキーインして下さい。**

設問 53　**正しくキーインされているかどうか，プログラムリストをすべてプリンタに印字して確認して下さい。**

【プログラム印字例】

```
10 ' ｼﾞｭｳｼｮﾛｸ ｻｸｾｲ ﾌﾟﾛｸﾞﾗﾑ
20 '
30 ' ｼﾞｭﾝﾋﾞ ﾙｰﾁﾝ
40 PNM$="1. ﾅﾏｴ     : &"+SPACE$(13)+"&"
50 PJU$="2. ｼﾞｭｳｼｮ : &"+SPACE$(28)+"&"
60 PTL$="3. Tel-No  : &"+SPACE$(10)+"&"
70 CLS
80 OPEN "2:JU-F" FOR OUTPUT AS #1
90 ' ﾃﾞｰﾀ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
100 PRINT "***** ﾃﾞｰﾀ ﾆｭｳﾘｮｸ *****":PRINT
110 GOSUB 500
120 IF NM$="END" OR NM$="end" THEN CLOSE#1:END
130 GOSUB 600:GOSUB 700:PRINT
```
（つづく）

```
140 ′ ﾃﾞｰﾀ ﾁｪｯｸ ﾙｰﾁﾝ
150 PRINT USING PNM$;NM$
160 PRINT USING PJU$;JUSHO$
170 PRINT USING PTL$;TEL$:PRINT
180 PRINT TAB(3);"ﾃｲｾｲ ｱﾘ [1-3]"
190 PRINT TAB(8);"ﾅｼ [RET] ";:INPUT C:PRINT
200 IF C<0 OR C>3 THEN PRINT "ｴﾗｰ!!":GOTO 180
210 IF C=0 THEN 240
220 ON C GOSUB 500,600,700:PRINT
230 GOTO 150
240 WRITE#1,NM$,JUSHO$,TEL$:PRINT
250 GOTO 100
500 ′ ﾅﾏｴ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
510 INPUT "ﾅﾏｴ = ";NM$
520 RETURN
600 ′ ｼﾞｭｳｼｮ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
610 INPUT "ｼﾞｭｳｼｮ = ";JUSHO$
620 RETURN
700 ′ Tel-No ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
710 INPUT "Tel-No = ";TEL$
720 RETURN
```

設問 54　正しくキーインできていたら，次のデータを使ってプログラムを実行して下さい。

（入力するデータ）

名 前	住 所	電話番号
ミト　ケイイチ	トウキョウト　シンジュクク　キタシンジュク　1-5-2	03-363-7451
オオツ　トシユキ	ナゴヤシ　アツタク　タイホウ　4-19-14	052-681-9500
マツヤマ　トシエ	トウキョウト　スギナミク　ミヤマエ　4-5-24	03-333-6451
チバ　ヤスヒロ	オオサカシ　ヒガシナリク　ナカモト　1-5-21	06-974-2164
ヤマグチ　ミチヨ	ヨコハマシ　ミドリク　エダキタ　2-2-8	045-911-4214

【 実行結果例 】
（画面）

```
***** ﾃﾞｰﾀ ﾆｭｳﾘｮｸ *****

ﾅﾏｴ = ? ﾐﾄ ｹｲｲﾁ
ｼﾞｭｳｼｮ = ? ﾄｳｷｮｳﾄ ｼﾝｼﾞｭｸｸ ｷﾀｼﾝｼﾞｭｸ 1-5-2
Tel-No = ? 03-363-7451

1. ﾅﾏｴ     : ﾐﾄ ｹｲｲﾁ
2. ｼﾞｭｳｼｮ : ﾄｳｷｮｳﾄ ｼﾝｼﾞｭｸｸ ｷﾀｼﾝｼﾞｭｸ 1-5-2
3. Tel-No : 03-363-7451
```

（つづく）

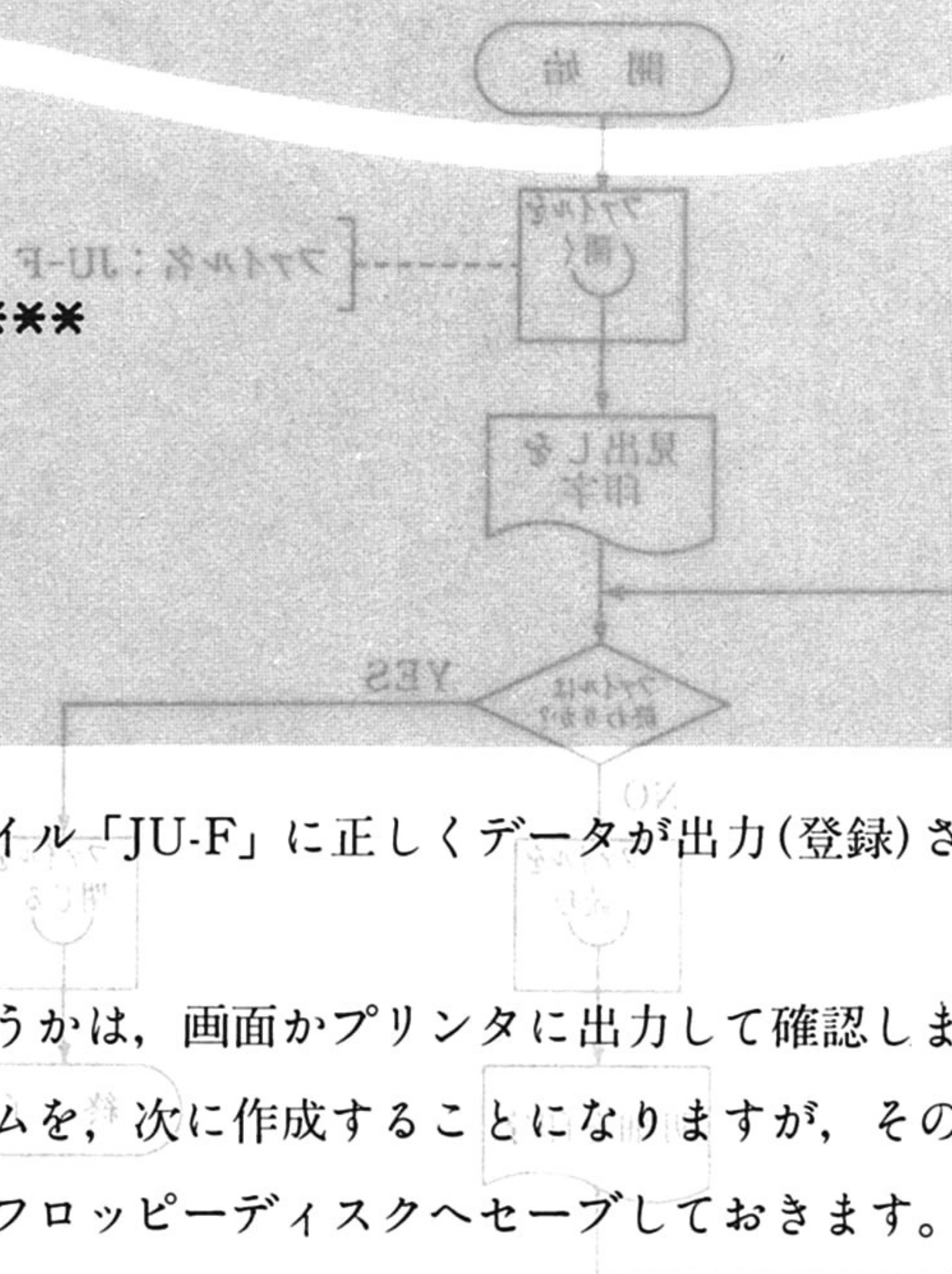
```
    テイセイ アリ [1-3]
          ナシ [RET] ?

*****  デ゛ータ ニュウリョク  *****

    テイセイ アリ [1-3]
          ナシ [RET] ?

*****  デ゛ータ ニュウリョク  *****

ナマエ = ? END
Ok
```

【 実行後の説明 】

・入力データに訂正があったときは，次のようになります。

```
*****  デ゛ータ ニュウリョク  *****

ナマエ = ? トミタ ケンイチ
ジ゛ュウショ = ? トウキョウト シンジ゛ュクク キタシンジ゛ュク 1-5-2
Tel-No = ? 03-396-7421

1. ナマエ     : トミタ ケンイチ
2. ジ゛ュウショ : トウキョウト シンジ゛ュクク キタシンジ゛ュク 1-5-2
3. Tel-No   : 03-396-7421

    テイセイ アリ [1-3]
          ナシ [RET] ? 1

ナマエ = ? ミト ケイイチ

1. ナマエ     : ミト ケイイチ
2. ジ゛ュウショ : トウキョウト シンジ゛ュクク キタシンジ゛ュク 1-5-2
3. Tel-No   : 03-396-7421

    テイセイ アリ [1-3]
          ナシ [RET] ? 3

Tel-No = ? 03-363-7451

1. ナマエ     : ミト ケイイチ
2. ジ゛ュウショ : トウキョウト シンジ゛ュクク キタシンジ゛ュク 1-5-2
3. Tel-No   : 03-363-7451

    テイセイ アリ [1-3]
          ナシ [RET] ?
```

(つづく)

```
***** デ゛ータ ニュウリョク *****

    テイセイ アリ [1-3]
         ナシ [RET] ?

***** デ゛ータ ニュウリョク *****

ナマエ = ? END
Ok
```

・このままでは，シーケンシャルファイル「JU-F」に正しくデータが出力(登録)されているかどうかわかりません。

・データが正しく登録されているかどうかは，画面かプリンタに出力して確認します。

・そこで，データを出力するプログラムを，次に作成することになりますが，その前に，今メモリーに記憶しているプログラムをフロッピーディスクへセーブしておきます。

設問 55　今メモリーに記憶しているプログラムを，ドライブ2のフロッピーディスクへ「JUSHO.1」という名前でセーブして下さい。

【 実行結果例 】

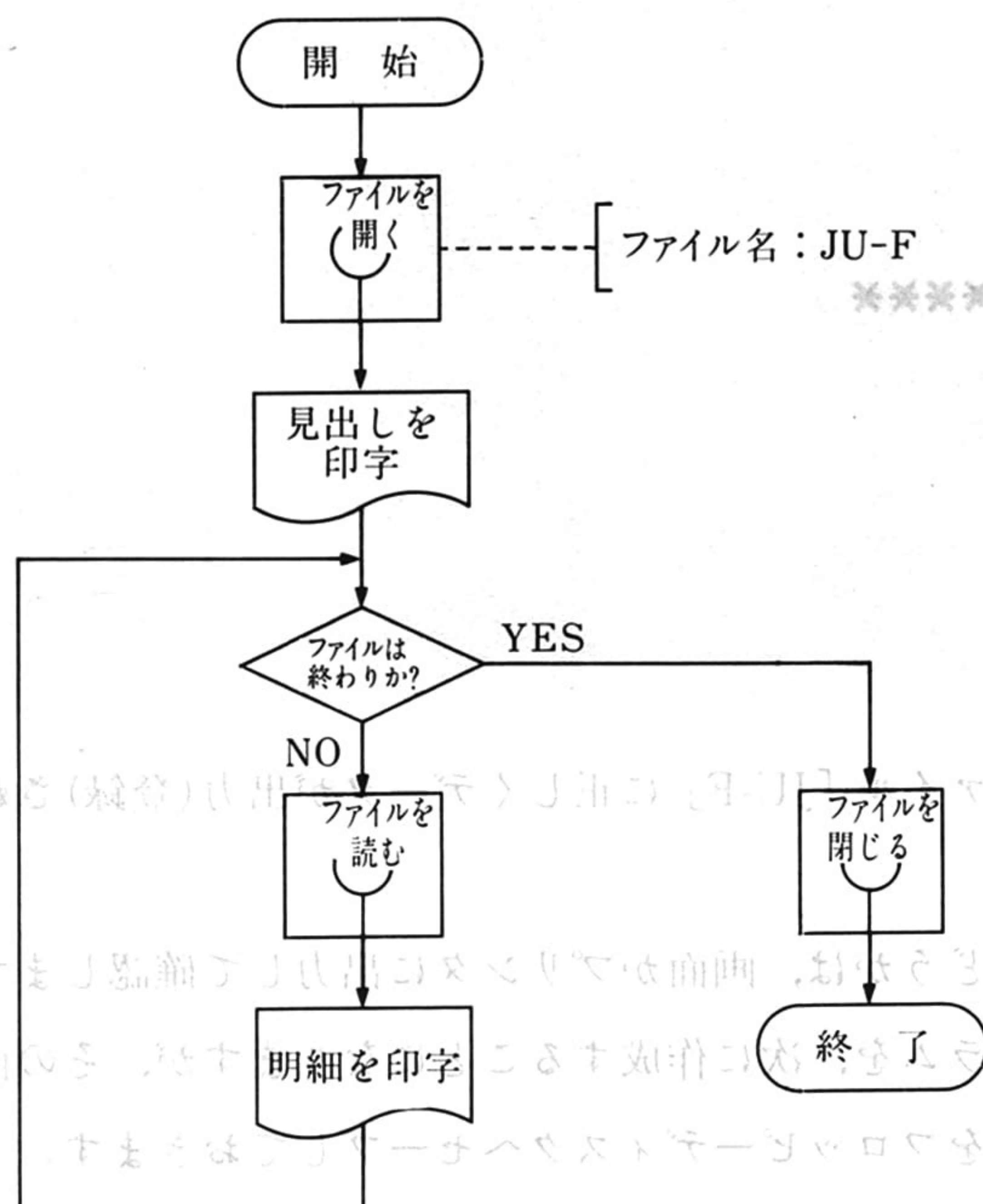
```
save "2:JUSHO.1"
Ok
```

6.2.4 シーケンシャルファイルの読み方

シーケンシャルファイル「JU-F」に登録したデータを読み込んで，これを編集したうえでプリンタに印字するには「ファイル入力プログラム」が必要になります。

そこで，6.2.1 (2)のプログラム作成の条件を満たすようなプログラムのフローチャート例を次に示します。

【 フローチャート例 】

開　始
ファイルを開く ---- ファイル名：JU-F
見出しを印字
ファイルは終わりか?　YES　NO
ファイルを読む
明細を印字
ファイルを閉じる
終　了

ここで示したフローチャート例に従って処理の手順を一つ一つ命令に置き換えると次のようなプログラムになります。

【 プログラム例 】

```
10 ' ｼﾞｭｳｼｮﾛｸ ｲﾝｼﾞ ﾌﾟﾛｸﾞﾗﾑ
20 LPNM$="&"+SPACE$(13)+"&"
30 LPJU$="&"+SPACE$(28)+"&"
40 LPTL$="&"+SPACE$(10)+"&"
50 OPEN "2:JU-F" FOR INPUT AS #1
60 LPRINT TAB(18);"****** ｼﾞｭｳｼｮﾛｸ ******"
70 LPRINT "ﾅﾏｴ";TAB(17);"ｼﾞｭｳｼｮ";TAB(49);"Tel-No"
80 LPRINT
90 IF EOF(1) THEN CLOSE#1:END
100 INPUT#1,NM$,JUSHO$,TEL$
110 LPRINT USING LPNM$;NM$;
120 LPRINT TAB(17);
130 LPRINT USING LPJU$;JUSHO$;
140 LPRINT TAB(49);
150 LPRINT USING LPTL$;TEL$
160 GOTO 90
```

【 プログラム例の説明 】

・ドライブ2のフロッピーディスクの「JU-F」という名前のシーケンシャルファイルからデータを読み込んで，編集してプリンタに印字しています。

・行番号20～40では，編集文字列を定義しています。
・行番号50では，ドライブ2のフロッピーディスクの「JU-F」という名前のシーケンシャルファイルを，INPUTモードでファイル番号「1」にオープンしています。
・行番号60～80では，見出しを印字しています。
・行番号90では，ファイル番号「1」のシーケンシャルファイルの読み込みが終了していれば，そのファイルを閉じて処理を終了しています。
・行番号100では，ファイル番号「1」のファイルのデータを，変数NM＄，JUSHO＄，TEL＄に読み込んでいます。変数NM＄には名前が，変数JUSHO＄には住所が，変数TEL＄には電話番号がそれぞれ入ります。
・行番号110～150では，プリンタに名前，住所，電話番号を編集して印字しています。

設問 56　メモリーをクリアした後，例で示したプログラムをキーインして下さい。

設問 57　正しくキーインされているかどうか，プログラムリストをすべて画面に表示して下さい。

【 プログラム表示例 】

```
list
10 ' ｼﾞｭｳｼｮﾛｸ ｲﾝｼﾞ ﾌﾟﾛｸﾞﾗﾑ
20 LPNM$="&"+SPACE$(13)+"&"
30 LPJU$="&"+SPACE$(28)+"&"
40 LPTL$="&"+SPACE$(10)+"&"
50 OPEN "2:JU-F" FOR INPUT AS #1
60 LPRINT TAB(18);"****** ｼﾞｭｳｼｮﾛｸ ******"
70 LPRINT "ﾅﾏｴ";TAB(17);"ｼﾞｭｳｼｮ";TAB(49);"Tel-No"
80 LPRINT
90 IF EOF(1) THEN CLOSE#1:END
100 INPUT#1,NM$,JUSHO$,TEL$
110 LPRINT USING LPNM$;NM$;
120 LPRINT TAB(17);
130 LPRINT USING LPJU$;JUSHO$;
140 LPRINT TAB(49);
150 LPRINT USING LPTL$;TEL$
160 GOTO 90
Ok
```

プログラムをメモリーに入力し終わったら，実際にプログラムを動かしてみましょう。

設問 58　今メモリーに記憶しているプログラムを実行して下さい。

【 実行結果例 】

```
                    ****** ｼﾞｭｳｼｮﾛｸ ******
ﾅﾏｴ              ｼﾞｭｳｼｮ                              Tel-No

ﾐﾄ ｹｲｲﾁ          ﾄｳｷｮｳﾄ ｼﾝｼﾞｭｸｸ ｷﾀｼﾝｼﾞｭｸ 1-5-2      03-363-7451
ｵｵﾂ ﾄｼﾕｷ         ﾅｺﾞﾔｼ ｱﾂﾀｸ ﾀｲﾎｳ 4-19-14            052-681-9500
ﾏﾂﾔﾏ ﾄｼｴ         ﾄｳｷｮｳﾄ ｽｷﾞﾅﾐｸ ﾐﾔﾏｴ 4-5-24          03-333-6451
ﾁﾊﾞ ﾔｽﾋﾛ         ｵｵｻｶｼ ﾋｶﾞｼﾅﾘｸ ﾅｶﾓﾄ 1-5-21          06-974-2164
ﾔﾏｸﾞﾁ ﾐﾁﾖ        ﾖｺﾊﾏｼ ﾐﾄﾞﾘｸ ｴﾀﾞｷﾀ 2-2-8            045-911-4214
```

【 実行後の説明 】

・名前の印字は15文字までに編集しています。

・住所の印字は30文字までに編集しています。

・電話番号の印字は12文字までに編集しています。

設問 59　正しく実行できていたら，今メモリーに記憶しているプログラムを，ドライブ2のフロッピーディスクへ「JUSHO.2」という名前でセーブして下さい。

【 実行結果例 】

```
save "2:JUSHO.2"
Ok
```

6.2.5　シーケンシャルファイルへデータを追加するには

今までは，シーケンシャルファイルを新たに作成するOPEN命令のOUTPUTモードと，作成されたシーケンシャルファイルのデータをメモリーに読み込むINPUTモードを学習してきました。しかしこのほかにも，既に作成しているシーケンシャルファイルを壊さずに，データを追加する場合が生じてきます。

ここでは，シーケンシャルファイルへデータを追加する方法を学習します。

(1)　データを追加する

シーケンシャルファイルへデータを追加するには，OPEN命令のAPPENDモードを使います。そこで，ファイルへデータを追加することができるプログラムのフローチャートが必要になるわけですが，そのフローチャートおよびプログラムを利用することにします。

設問 60　ドライブ2のフロッピーディスクにセーブしている「JUSHO.1」というプログラムを，メモリーに読み込んで（ロードして）下さい。

【 実行結果例 】

```
load "2:JUSHO.1"
Ok
```

【 実行後の説明 】

・これで，メモリーにファイル作成プログラムが読み込まれました。

設問 61　今メモリーにロードしたプログラムのうち，先頭から行番100までを画面に表示して下さい。

【 プログラム表示例 】

```
list -100
10 ' ｼﾞｭｳｼｮﾛｸ ｻｸｾｲ ﾌﾟﾛｸﾞﾗﾑ
20 '
30 ' ｼﾞｭﾝﾋﾞ ﾙｰﾁﾝ
40 PNM$="1. ﾅﾏｴ     : &"+SPACE$(13)+"&"
50 PJU$="2. ｼﾞｭｳｼｮ  : &"+SPACE$(28)+"&"
60 PTL$="3. Tel-No  : &"+SPACE$(10)+"&"
70 CLS
80 OPEN "2:JU-F" FOR OUTPUT AS #1
90 ' ﾃﾞｰﾀ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
100 PRINT "***** ﾃﾞｰﾀ ﾆｭｳﾘｮｸ *****":PRINT
Ok
```

【 表示後の説明 】

・行番号80で，ドライブ2の「JU-F」というファイルをOUTPUTモードでオープンしています。

・このOPEN命令を，OUTPUTモードからAPPENDモードへ変更して，データの追加ができるようにします。

設問 62　行番号80のOPEN命令をOUTPUTモードからAPPENDモードへ変更して下さい。

【修正例】

```
80 OPEN "2:JU-F" FOR OUTPUT AS #1
```

⇩

```
80 OPEN "2:JU-F" FOR APPEND AS #1
```

設問 63　正しく修正されていけるかどうか，先頭から行番号100までのプログラムリストを画面に表示して確認して下さい。

【プログラム表示例】

```
list -100
10 ' ｼﾞｭｳｼｮﾛｸ ｻｸｾｲ ﾌﾟﾛｸﾞﾗﾑ
20 '
30 ' ｼﾞｭﾝﾋﾞ ﾙｰﾁﾝ
40 PNM$="1. ﾅﾏｴ      : &"+SPACE$(13)+"&"
50 PJU$="2. ｼﾞｭｳｼｮ : &"+SPACE$(28)+"&"
60 PTL$="3. Tel-No : &"+SPACE$(10)+"&"
70 CLS
80 OPEN "2:JU-F" FOR APPEND AS #1
90 ' ﾃﾞｰﾀ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
100 PRINT "***** ﾃﾞｰﾀ ﾆｭｳﾘｮｸ *****":PRINT
Ok
```

設問 64　正しく修正されていたら，プログラムリストをすべてプリンタへ印字して下さい。

【プログラム印字例】

```
10 ' ｼﾞｭｳｼｮﾛｸ ｻｸｾｲ ﾌﾟﾛｸﾞﾗﾑ
20 '
30 ' ｼﾞｭﾝﾋﾞ ﾙｰﾁﾝ
40 PNM$="1. ﾅﾏｴ      : &"+SPACE$(13)+"&"
50 PJU$="2. ｼﾞｭｳｼｮ : &"+SPACE$(28)+"&"
60 PTL$="3. Tel-No : &"+SPACE$(10)+"&"
70 CLS
80 OPEN "2:JU-F" FOR APPEND AS #1
90 ' ﾃﾞｰﾀ ﾆｭｳﾘｮｸ ﾙｰﾁﾝ
100 PRINT "***** ﾃﾞｰﾀ ﾆｭｳﾘｮｸ *****":PRINT
110 GOSUB 500
```

（つづく）

```
120 IF NM$="END" OR NM$="end" THEN CLOSE#1:END
130 GOSUB 600:GOSUB 700:PRINT
140 ' デ゛ータ チェック ルーチン
150 PRINT USING PNM$;NM$
160 PRINT USING PJU$;JUSHO$
170 PRINT USING PTL$;TEL$:PRINT
180 PRINT TAB(3);"テイセイ アリ [1-3]"
190 PRINT TAB(8);"ナシ [RET] ";:INPUT C:PRINT
200 IF C<0 OR C>3 THEN PRINT "エラー!!":GOTO 180
210 IF C=0 THEN 240
220 ON C GOSUB 500,600,700:PRINT
230 GOTO 150
240 WRITE#1,NM$,JUSHO$,TEL$:PRINT
250 GOTO 100
500 ' ナマエ ニュウリョク ルーチン
510 INPUT "ナマエ = ";NM$
520 RETURN
600 ' ジ゛ュウショ ニュウリョク ルーチン
610 INPUT "ジ゛ュウショ = ";JUSHO$
620 RETURN
700 ' Tel-No ニュウリョク ルーチン
710 INPUT "Tel-No = ";TEL$
720 RETURN
```

設問 65　次のデータでプログラムを実行して下さい。

（入力するデータ）

名前	住所	電話番号
オカヤマ　ユウコ	チョウフシ シバサキチョウ 42-1	0424-88-7410
ナガノ　カズヨシ	カワサキシ タマク イクタ 8-11-2	044-933-0745
アキタ　ヒデオ	ニイガタシ ベンテン 2-3-13	0252-41-1181
ミヤザキ　ツカサ	フクオカシ チュウオウク ハルヨシ 1-11-18	092-711-0401
フクイ　チエミ	サッポロシ シロイシク キクスイ 6ジョウ 3-4-28	011-831-5511

【 実行結果例 】

```
***** デ゛ータ ニュウリョク *****

ナマエ = ? オカヤマ ユウコ
ジ゛ュウショ = ? チョウフシ シバ゛サキチョウ 42-1
Tel-No = ? 0424-88-7410

1. ナマエ    : オカヤマ ユウコ
2. ジ゛ュウショ : チョウフシ シバ゛サキチョウ 42-1
3. Tel-No  : 0424-88-7410
```

（つづく）

```
    テイセイ アリ [1-3]
           ナシ [RET] ?

***** データ ニュウリョク *****
```

```
    テイセイ アリ [1-3]
           ナシ [RET] ?

***** データ ニュウリョク *****

ナマエ = ? END
Ok
```

【 実行後の説明 】

・ドライブ2の「JU-F」というシーケンシャルファイルの5件のレコードの後に，更に5件のレコードが追加登録されました。

・データが正しく登録されているかどうか，ファイル入力プログラムを使ってデータをプリンタに印字して確認します。

設問 66　今メモリーに記憶しているプログラムを，ドライブ2のフロッピーディスクへ「JU-SHO.3」という名前でセーブして下さい。

【 実行結果例 】

```
save "2:JUSHO.3"
Ok
```

設問 67　ドライブ2のフロッピーディスクの「JUSHO.2」というプログラムをロードした後，実行して下さい。

【 実行結果例 】

(画面)

```
load "2:JUSHO.2"
Ok
run
Ok
```

(プリンタ)

```
                        ****** ｼﾞｭｳｼｮﾛｸ ******
ﾅﾏｴ                    ｼﾞｭｳｼｮ                              Tel-No

ﾐﾄ ｹｲｲﾁ                ﾄｳｷｮｳﾄ ｼﾝｼﾞｭｸｸ ｷﾀｼﾝｼﾞｭｸ 1-5-2      03-363-7451
ｵｵﾂ ﾄｼﾕｷ               ﾅｺﾞﾔｼ ｱﾂﾀｸ ﾀｲﾎｳ 4-19-14             052-681-9500
ﾏﾂﾔﾏ ﾄｼｴ               ﾄｳｷｮｳﾄ ｽｷﾞﾅﾐｸ ﾐﾔﾏｴ 4-5-24          03-333-6451
ﾁﾊﾞ ﾔｽﾋﾛ               ｵｵｻｶｼ ﾋｶﾞｼﾅﾘｸ ﾅｶﾓﾄ 1-5-21           06-974-2164
ﾔﾏｸﾞﾁ ﾐﾁﾖ              ﾖｺﾊﾏｼ ﾐﾄﾞﾘｸ ｴﾀﾞｷﾀ 2-2-8             045-911-4214
ｵｶﾔﾏ ﾕｳｺ               ﾁｮｳﾌｼ ｼﾊﾞｻｷﾁｮｳ 42-1                0424-88-7410
ﾅｶﾞﾉ ｶｽﾞﾖｼ             ｶﾜｻｷｼ ﾀﾏｸ ｲｸﾀ 8-11-2               044-933-0745
ｱｷﾀ ﾋﾃﾞｵ               ﾆｲｶﾞﾀｼ ﾍﾞﾝﾃﾝ 2-3-13                0252-41-1181
ﾐﾔｻﾞｷ ﾂｶｻ              ﾌｸｵｶｼ ﾁｭｳｵｳｸ ﾊﾙﾖｼ 1-11-18           092-711-0401
ﾌｸｲ ﾁｴﾐ                ｻｯﾎﾟﾛｼ ｼﾛｲｼｸ ｷｸｽｲ 6ｼﾞｮｳ 3-4-28    011-831-5511
```

【 実行後の説明 】

・ファイル作成プログラム「JUSHO.1」で登録した5件のデータの後に，更に5件のデータが追加されていることが分かります。

7章　関数

このテキストでは，今までに「TAB」，「SEARCH」，「SPACE$」，「EOF」の四つの関数を学習しました。

この章では，前出の四つのほかに8章と9章で新たに出てくる関数を説明します。

この章で説明する関数は，次の8種類です。

（数値関数）

- INT
- RND

（文字関数）

- STRING$
- VAL
- LEFT$
- RIGHT$
- CHR$
- INKEY$

7.1　数値関数

数値関数は，数値演算を行うための関数です。

7.1.1　INT関数

INT関数は，数値の小数部を省略して，その数値を超えない最大の整数を求めます。

（INT関数の説明）

〈一般形式〉

INT（数式）

- 数式の値を超えない最大の整数を求めます。
- 数式が正の値のときは，小数部の切り捨てが行われます。
- 数式が負の値のときは，その絶対値が切り上げになるように小数部の切り捨てが行われます。

〈書き方例〉

①　PRINT　INT(6.9)	画面に「6.9」の小数部を切り捨てた値「6」を表示します。
②　B=INT(−4.2)	変数Bに「−4.2」の絶対値を切り上げて小数部を切り捨てた値「−5」を代入します。
③　N=(−3.14159) PRINT　INT(N)	変数Nに代入されている値「−3.14159」の絶対値を切り上げて小数部を切り捨てた値「−4」を画面に表示します。

7.1.2 RND関数

RND関数は，乱数を発生させます。

(RND関数の説明)

<一般形式>

RND（数式）

・数式に与えられた条件で，乱数を発生させます。

・乱数は0.000001～0.999999の範囲で発生します。

・数式に与えられる条件には，次の３種類のものがあります。

数式	意　味
負の値	新しい乱数系列で乱数を発生します。
0	一つ前に発生した乱数の値を与えます。
正の値	次の乱数を発生します。

・数式が省略されたときは，正の値が与えられたものとして次の数を発生させます。

<書き方例>

①	PRINT　RND(－３)	新しい乱数系列で発生した乱数を画面に表示します。乱数系列が初期値ならば「0.308601」が表示されます。
②	A＝RND(０)	一つ前に発生した乱数の値を変数Aに代入します。
③	B＝RND(１)	次の乱数を発生させて変数Bに代入します。
④	N＝５ PRINT　RND(N)	次の乱数を発生させて画面に表示します。

7.2 文字関数

文字関数は，文字列を取り扱う関数です。

7.2.1 STRING$関数

STRING$関数は，同一文字を連続して与えます。

(STRING$関数の説明)

〈一般形式〉

STRING$（式，文字式または数式）

- 文字式または数式で表す文字を，式で指定した文字数だけで作り出します。
- 文字式として2桁以上の文字列を指定したときは，初めの1文字だけが利用されます。
- 数式で指定すると，この値を文字コードとみなし，これに対応する文字が利用されます。
- 数式に指定できる値の範囲は0～255です。

〈書き方例〉

①	A$=STRING$(10, "X")	「X」を10文字連続させて，変数A$に代入します。
②	B$="Y" PRINT STRING$(5, B$)	「Y」を5文字連続させて，画面に表示します。
③	C$="XYZ" M$=STRING$(8, M$)	「XYZ」の先頭の「X」を8文字連続させて，変数M$に代入します。
④	D=15：X=69 N$=STRING$(D, X)	文字コード「69」の文字「E」を15文字連続させて，変数N$に代入します。

7.2.2 VAL関数

VAL関数は，文字型で表された数字を数値型の値に変換します。

(VAL関数の説明)

〈一般形式〉

VAL（文字列）

- 文字列の表すデータを数値に変換します。
- 文字列の最初の文字が「＋」，「－」，「.」，「&」または数字のいずれでもなければ，結果は「0」になります。
- 文字列中にスペースがある場合，そのスペースを無視して変換します。

〈書き方例〉

①	PRINT　VAL("20")	文字型データ「20」を数値変換して数値の「20」にし，画面に表示します。
②	A＝VAL("－35.44")	文字型データ「－35.44」を数値変換して数値の「－35.44」にし，変数Aに代入します。
③	N$＝"95.6" B＝VAL(N$)	文字型データ「95.6」を数値変換して数値の「95.6」にし，変数Bに代入します。

7.2.3　LEFT$関数

LEFT$関数は，文字列の左側から指定した長さの文字列を取り出す関数です。

(LEFT$関数の説明)

〈一般形式〉

LEFT$（文字列，数式）

- 文字列の左側から数式で指定した長さの文字列を取り出します。
- 数式には，０～255の範囲の値を指定することができます。
- 数式に文字列の総文字数以上の値が指定されたときは，文字列のすべてを取り出します。
- 数式に「０」が指定されれば，ヌルストリングを与えます。

〈書き方例〉

① PRINT　LEFT$("ABCDEFG"，３)

「ABCDEFG」という文字列の左側から３文字分の「ABC」を取り出し，画面に表示します。

② A$＝LEFT$("トウキョウ"，４)

「トウキョウ」という文字列の左側から４文字分の「トウキョ」を取り出し，変数A$に代入します。

③ M$＝"オオサカ"：X＝２
B$＝LEFT$(M$，X)

「オオサカ」という文字列の左側から２文字分の「オオ」を取り出し，変数B$に代入します。

④ N$＝"フクオカ"：Y＝８
C$＝LEFT$(N$，Y)

「フクオカ」という文字列の左側から８文字分，つまりこの場合は全部を変数C$に代入します。

⑤ D$＝LEFT$("ナゴヤ"，０)

変数D$にヌルストリングを代入します。

7.2.4 RIGHT$関数

RIGHT$関数は，文字列の右側から指定した長さの文字列を取り出す関数です。

(RIGHT$関数の説明)

〈一般形式〉

RIGHT$（文字列，数式）

・文字列の右側から数式で指定した長さの文字列を取り出します。

・数式には，0～255の範囲の値を指定することができます。

・数式に文字列の総文字数以上の値が指定されたときは，文字列のすべてを取り出します。

・数式に「0」が指定されれば，ヌルストリングを与えます。

〈書き方例〉

① PRINT RIGHT$("ABCDEFG", 3)

「ABCDEFG」という文字列の右側から3文字分の「EFG」を取り出し，画面に表示します。

② A$=RIGHT$("トウキョウ", 4)

「トウキョウ」という文字列の右側から4文字分の「ウキョウ」を取り出し，変数A$に代入します。

③ M$="オオサカ":X=2

B$=RIGHT$(M$, X)

「オオサカ」という文字列の右側から2文字分の「サカ」を取り出し，変数B$に代入します。

④ N$="フクオカ":Y=8

C$=RIGHT$(N$, Y)

「フクオカ」という文字列の右側から8文字分，つまりこの場合は全部を変数C$に代入します。

⑤ D$=RIGHT$("ナゴヤ", 0)

変数D$にヌルストリングを代入します。

7.2.5 CHR$関数

PC-8801mkIIで使用できる文字には，すべて順番にコードが付いています。CHR$関数は，文字をその文字コードで表すときに使います。

(CHR$関数の説明)

〈一般形式〉

CHR$（数式）

・数式の値で示す文字コードの文字を求めます。

・数式には，０～255の範囲の値を指定することができます。

・実際に文字に対応するのは32～247の範囲で，それ以外は制御（コントロール）コードか未使用のコードになっています。(「付.3　キャラクタコード表を参照)

〈書き方例〉

①	PRINT　CHR$(65)	画面「65」という文字コードに対応する文字「A」を表示します。
②	X＝38 N$＝CHR$(X)	「38」という文字コードに対応する文字「&」を変数N$に代入します。
③	Y＝13 M$＝CHR$(Y)	「13」という文字コードに対応するコントロールコード「$^{C}_{R}$(リターンコード)」を変数M$に代入します。

7.2.6 INKEY$関数

INKEY$関数は，任意の文字を一文字だけ入力する関数です。

(INKEY$関数の説明)

〈一般形式〉

INKEY$

・キーが押されていればその文字を，キーが押されなければヌルストリングを入力します。

・CTRL ⊕ C と STOP 以外のキーは，すべて入力することができます。

・押されたキーに対応する文字は，画面には表示されません。

〈書き方例〉

```
10  A$=INKEY$
20  IF A$="" THEN  10
30  PRINT  A$
```

行番号10の命令文を実行したときに，キーが押されていればその文字がA$へ，キーが押されていなければヌルストリングがA$へ入力されます。

したがって，この例は何かのキーが押されるまで行番号10～20の実行を繰り返します。

8章　漢字

PC-8801mkIIでは，画面プリンタに漢字を出力することができます。そのため，通信文書や報告書などを，漢字を使って見やすいものを作成することができます。

8.1　漢字コード表の見方

PC-8801mkIIや漢字プリンタ（PC-8822）は，漢字が登録されている「漢字ROM」というメモリーを内蔵していて，その「漢字ROM」を使って漢字を画面に表示したり，プリンタに印字したりすることができます。

出力できる漢字の種類は，「JIS第１水準の漢字2965文字」と「非漢字約700文字」です。

また，PC-8801mkIIで使用する一つ一つの漢字には，漢字コードが付けられています。したがって，漢字を使う場合には必ず漢字コードが必要になります。そこで，漢字の学習をする前に「付.4　漢字コード表」の見方について説明しておきます。

漢字コード表は，原則として音読みで調べることになるので，たとえば「雨」という漢字のコードを調べる場合は，次の図のように漢字コード表の ウ の欄を探します。そして，この表の中の「雨」という漢字が書かれている位置を調べます。

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ウ	312							右	宇	烏	羽	迂	雨	卯	鵜	窺	丑
	313	碓	臼	渦	嘘	唄		蔚	鰻	姥	厩	浦	瓜	閏	噂	云	運
	314	雲															

B　312　雨

「雨」は"312B"で表示されます。

これで，「雨」という漢字の位置は，"312"と"B"で示されることがわかります。つまり，"312B"という16進コードが「雨」の漢字コードというわけです。

それでは，次に「安い」という文字の漢字コードを調べて，少し練習してみます。まず「安」ですが，これは音読みで「アン」なのでコード表の ア の欄を探します。下に示すように「安」という漢字の位置は"304"と"２"で示されているので，漢字コードは"3042"ということになります。

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ア	302		亜	唖	娃	阿	哀	愛	挨	姶	逢	葵	茜	穐	悪	握	渥
	303	旭	葦	芦	鯵	梓	圧	斡	扱	宛	姐	虻	飴	絢	綾	鮎	或
	304	粟	袷	安	庵	按	暗	案	闇	鞍	杏						

2　304　安

「安」は"3042"で示されます。

次に，「い」ですが，これはコード表の ひらがな の欄を探します。下に示すように「い」という字の位置は“242”と“4”で示されているので，漢字コードは“2424”ということになります。

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ひらがな	242		ぁ	あ	ぃ	い		う	ぇ	え	ぉ	お	か	が	き	ぎ	く
	243	ぐ	け	げ	こ	ご	さ	ざ	し	じ	す	ず	せ	ぜ	そ	ぞ	た
	244	だ	ち	ぢ	っ	つ	づ	て	で	と	ど	な	に	ぬ	ね	の	は
	245	ば	ぱ	ひ	び	ぴ	ふ	ぶ	ぷ	へ	べ	ぺ	ほ	ぼ	ぽ	ま	み
	246	む	め	も	ゃ	や	ゅ	ゆ	ょ	よ	ら	り	る	れ	ろ	ゎ	わ
	247	ゐ	ゑ	を	ん												

4

242 い

「い」は“2424”で示されます。

したがって，「安い」という文字の漢字コードは“3042”と“2424”ということになります。このように，一つ一つの漢字についているコードは，漢字コード表を使って調べます。

〈一口メモ〉

パラメータ

コマンド，命令，関数で特定の意味を表すために指定する値を「パラメータ」と言います。例えば，「LIST 50-100」や「CONSOLE 15，5，0，1」の「50-100」や「15，5，0，1」などのことを総称して「パラメータ」と言います。

8.2 漢字の表示

ここでは，漢字を表示するにはどの命令を使ってどんな方法で行うかについて学習します。

8.2.1 漢字を画面に表示するには

漢字を画面に表示するには,「PUT」命令を使って，指定した漢字コードの漢字をグラフィック画面へ出力します。

PUT命令とグラフィック画面のモードを設定する「SCREEN」命令について説明します。

(SCREEN命令の説明)

〈一般形式〉

SCREEN 画面モード，画面スイッチ，アクティブページ，ディスプレイページ

・グラフィック画面の使い方を設定します。

・画面モードには0～2の範囲の値を指定し，画面を次のいずれかの状態にします。

0………カラーモード　　　　(640×200ドット)

1………白黒モード　　　　　(640×200ドット)

2………高分解能白黒モード(640×400ドット)

・カラーモードを指定すると，グラフィック画面に描く漢字や図形に色を付けることができます。

・白黒モードと高分解能白黒モードでは,漢字や図形に色を付けることはできません。

・画面モードの初期値は「0」で，カラーモードになっています。

・画面モードを省略すると，既に指定されているモードがそのまま維持されます。

・画面スイッチ，アクティブページ，ディスプレイページの説明は，ここでは省略します。(詳しくは下巻で説明します)

〈書き方例〉

① SCREEN 0　　　グラフィック画面をカラーモードにします。

② SCREEN 1　　　グラフィック画面を白黒モードにします。

③ SCREEN 2　　　グラフィック画面を高分解能白黒モードにします。

(PUT命令の説明)

〈一般形式〉

PUT (X座標，Y座標)，KANJI (漢字コード)，条件，フォアグラウンドカラー，バックグラウンドカラー

・指定した漢字コードに該当する漢字を座標(X，Y)の位置に16×16ドットの大きさで表示します。

・漢字コードは通常16進数で指定します。

・この命令一回の実行で表示できる漢字は一文字です。

・条件には，次のものが指定できます。

PSET———— 漢字をそのまま表示します。

PRESET———— 白黒モードと高分解能白黒モードのときは，漢字をリバース（白黒反転）して表示します。カラーモードのときは，フォアグラウンドカラーおよびバックグラウンドカラーの値をそれぞれ「7－(表示する色のパレット番号)」の状態にして，漢字を表示します。

OR———— 指定された漢字と既に表示されている漢字とを，重ね合わせて表示します。ただし，カラーモードのときは，二つの漢字の色が違うと重なった部分の色が合成されて表示されます。

AND———— 指定された漢字と既に表示されている漢字との，重なった部分だけを表示します。ただし，カラーモードのときは，二つの漢字の色が違うと色が変わったり消えたりします。

XOR———— 白黒モードと高分解能白黒モードのときは，指定された漢字と既に表示されている漢字とを重ね合わせ，しかも重なった部分は消して表示します。カラーモードのときは，重ならない部分はそのまま表示しますが，重った部分は色を変えて表示します。

・条件を省略すると，「XOR」を指定したものとみなします。

・フォアグラウンドカラーには，表示する漢字の色をパレット番号（0～7）で指定します。白黒モードのときは，奇数(1，3，5，7)にすると白，偶数(0，2，4，6)にすると黒になります。

・バックグラウンドカラーには，表示する漢字の背景の色をパレット番号（0～7）で指定します。白黒モードのときは，奇数(1，3，5，7)にすると白，偶数(0，2，4，6）にすると黒になります。

・フォアグラウンドカラーとバックグラウンドカラーは，どちらか一方だけを省略することはできません。

・フォアグラウンドカラーおよびバックグラウンドカラーを省略すると，「7，0」が指定されたものとみなします。

・フォアグラウンドカラーやバックグラウンドカラーに指定するパレット番号に対応する色は，任意に変更することができますが，初期値は次のとおりです。

パレット番号	色
0	黒
1	青
2	赤
3	紫
4	緑
5	水色
6	黄色
7	白

〈書き方例〉

① SCREEN 1

PUT (200, 50), KANJI (&H4A73), PSET

「報」という漢字を白黒モードの画面に白色で表示します。

② SCREES 0

PUT (220, 50), KANJI (&H3970), PSET, 7, 1

「告」という漢字を青色の背景に白色で表示します。

③ SCREEN 0

A＝&H3D71

PUT (240, 50), KANJI (&H3D71), PRESET, 6, 1

「書」という漢字を黄色（「7－1」の色）の背景に青色（「7－6」の色）で表示します。

〈一口メモ〉

ドット

パソコンでは，一般に画面に表示される文字や図形やプリンタに印字される文字などはすべて細かな点の集まりで表されていて，この一つ一つの点のことを「ドット」と呼んでいます。

8.2.2　プログラム例(6)————利息の大小を比較するプログラム

それでは，実際に漢字を表示するプログラムを作ってみることにしましょう。

ここでは，元金と年数を入力し，「当座預金」，「普通預金」，「通知預金」，「期日指定定期預金」のn年後の元利合計を算出し表示するプログラムを作成します。画面へ表示するときに見出しと4種類の預金名を漢字にします。

(1)　プログラム作成の条件

「利息比較一覧プログラム」を作成するときの条件を設定すると，次のとおりです。

【 作成の条件 】

- 元金と年数をキーボードから入力し，「当座預金」，「普通預金」，「通知預金」，「期日指定定期預金」の4種類の預金のn年後の元利合計を計算して，画面の所定の位置に表示します。
- 表示する画面のサイズは，グラフィック画面は640×200のカラーモードで，テキスト画面は80×20文字にします。
- 漢字の表示は，漢字コードをDATA文からいったん配列に記憶させ，それを所定の位置に一度に表示します。
- 漢字の表示にはサブルーチンを使い，表示開始位置とフォアグラウンドカラーをパラメータとして漢字表示サブルーチンに引き渡します。
- 表示する漢字とその表示開始位置およびフォアグラウンドカラーは，それぞれ次のようになります。

表示する漢字	表示開始位置	フォアグラウンドカラー
利息比較一覧	(160,1)	赤
元金	(340,25)	水色
年後	(474,45)	〃
当 座 預 金	(　0,65)	緑
普 通 預 金	(　0,85)	〃
通 知 預 金	(　0,105)	〃
期日指定定期預金	(　0,125)	〃

- 表示する漢字は，バックグラウンドはすべて黒で，「0」を指定します。
- 表示する漢字の漢字コードを，次に示します。なお，漢字コードは16進数で表現しています。

漢字	漢字コード
利	4D78
息	4229
比	4866
較	3353

漢字	漢字コード
一	306C
覧	4D77
	2121
	2121

漢字	漢字コード
元	3835
金	3662
	2121
	2121

漢字	漢字コード
	2121
	2121
	2121
	2121

漢字	漢字コード
年	472F
後	3865
	2121
	2121
	2121
	2121
	2121
	2121
	2121
当	4576

漢字	漢字コード
	2121
座	3A42
	2121
預	4D42
	2121
金	3662
	2121
普	4961
	2121
通	444C

漢字	漢字コード
	2121
預	4D42
	2121
金	3662
	2121
通	444C
	2121
知	434E
	2121
預	4D42

漢字	漢字コード
	2121
金	3662
期	347C
日	467C
指	3B58
定	446A
定	446A
期	347C
預	4D42
金	3662

（注）「2121」はスペース（空白）を表す漢字コードです。

・漢字の一回の表示は，スペースを含めて８文字の漢字を，横に20ドットずつずらしながら行います。

・テキスト画面は，第16行（17行目）から２行だけスクロールするようにして，この行をデータ入力用として使います。

・元金の入力には倍精度変数を，年数の入力には単精度変数を使います。

・元金の入力で ⏎ キーを押したとき（「０」が入力されたとき）に，画面のスクロール状態を初期値（０行から25行スクロール）に戻して処理を終わります。

・「当座預金」は，年利０％で計算します。n年後の元利合計は次のようになります。

　　元利合計＝元金＊$(1+0)^n$

・「普通預金」は，年利1.5％で計算します。n年後の元利合計は次のようになります。

　　元利合計＝元金＊$(1+0.015)^n$

・「通知預金」は，年利1.75％で計算します。n年後の元利合計は次のようになります。

　　元利合計＝元金＊$(1+0.0175)^n$

・「期日指定定期預金」は，年利5.5％の半年複利で計算します。n年後の元利合計は半年複利なので次のようになります。

　　元利合計＝元金＊$(1+0.055/2)^{2n}$

・プログラムを実行している間だけ，ファンクションキーの内容を表示しないようにします。

・画面のレイアウトは，次のようになります。

利息比較一覧

(160,1)

(340,25) 元金 = ¥¥##,###,### (474,45)

年後

(0,65) 当座預金 : ¥¥###,###,###

(0,85) 普通預金 : ¥¥###,###,###

(0,105) 通知預金 : ¥¥###,###,###

(0,125) 期日指定定期預金 : ¥¥###,###,###

ｶﾞﾝｷﾝ = ?

ﾅﾝﾈﾝｺﾞ = ?

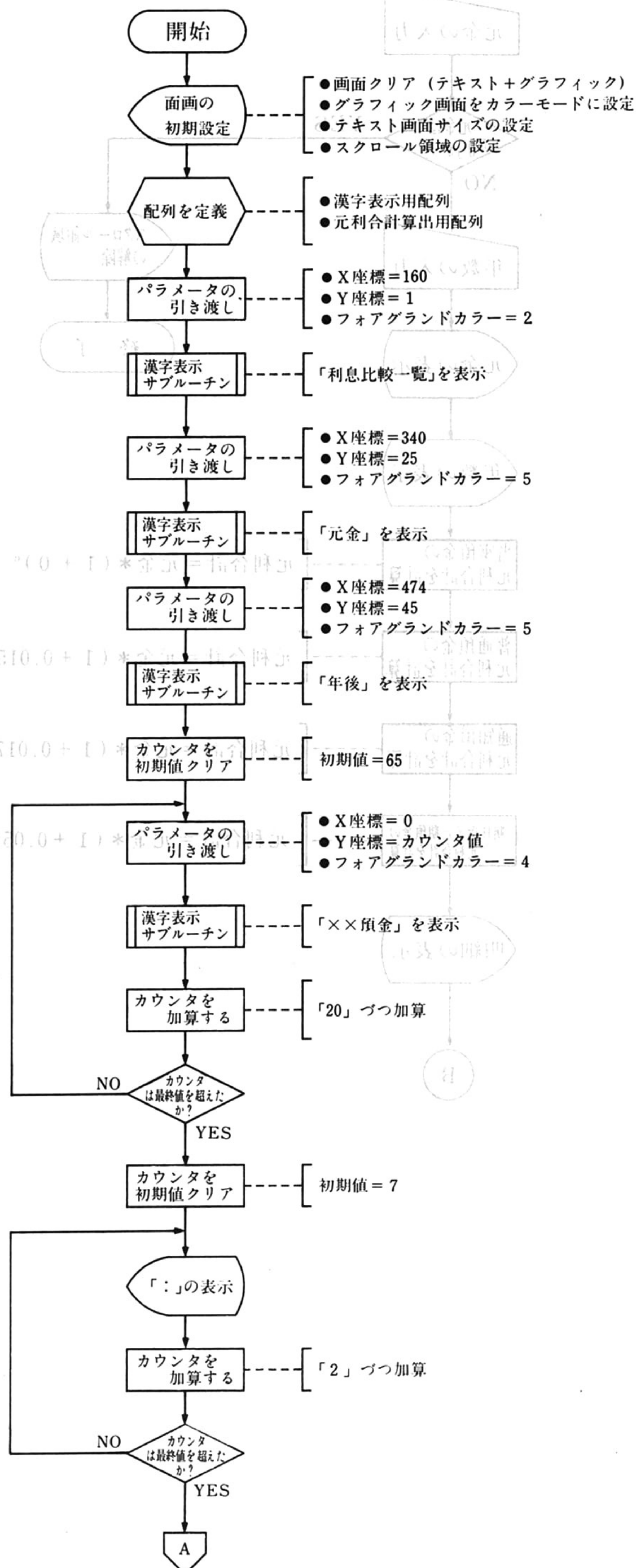
(注)・「21」,「23」,「50」,「55」のそれぞれの数字は，テキスト画面の表示文字位置を表します。

・「(160, 1)」,「(0, 65)」などの「( )」で囲まれた数字は，グラフィック画面での表示開始位置を表します。

（2）　フローチャートの作成

(1)で設定した作成条件を満たすための処理手順を考えて，次のようなフローチャートを作成します。

【 フローチャート 】

・メインルーチン

開始

画面の初期設定 ---- ●画面クリア（テキスト＋グラフィック） ●グラフィック画面をカラーモードに設定 ●テキスト画面サイズの設定 ●スクロール領域の設定

配列を定義 ---- ●漢字表示用配列 ●元利合計算出用配列

パラメータの引き渡し ---- ●X座標＝160 ●Y座標＝1 ●フォアグランドカラー＝2

漢字表示サブルーチン ---- 「利息比較一覧」を表示

パラメータの引き渡し ---- ●X座標＝340 ●Y座標＝25 ●フォアグランドカラー＝5

漢字表示サブルーチン ---- 「元金」を表示

パラメータの引き渡し ---- ●X座標＝474 ●Y座標＝45 ●フォアグランドカラー＝5

漢字表示サブルーチン ---- 「年後」を表示

カウンタを初期値クリア ---- 初期値＝65

パラメータの引き渡し ---- ●X座標＝0 ●Y座標＝カウンタ値 ●フォアグランドカラー＝4

漢字表示サブルーチン ---- 「××預金」を表示

カウンタを加算する ---- 「20」づつ加算

カウンタは最終値を超えたか？ NO / YES

カウンタを初期値クリア ---- 初期値＝7

「：」の表示

カウンタを加算する ---- 「2」づつ加算

カウンタは最終値を超えたか？ NO / YES

A

A

B

テキスト画面のクリア

元金の入力

元金は「0」か？ — YES → スクロール領域の解除 → 終了

NO

年数の入力

元金の表示

年数の表示

当座預金の元利合計を計算 ------ 元利合計＝元金＊（1＋0）^{n}

普通預金の元利合計を計算 ------ 元利合計＝元金＊（1＋0.015）^{n}

通知預金の元利合計を計算 ------ 元利合計＝元金＊（1＋0.0175）^{n}

期日指定定期預金の元利合計を計算 ------ 元利合計＝元金＊（1＋0.055／2）^{2n}

明細の表示

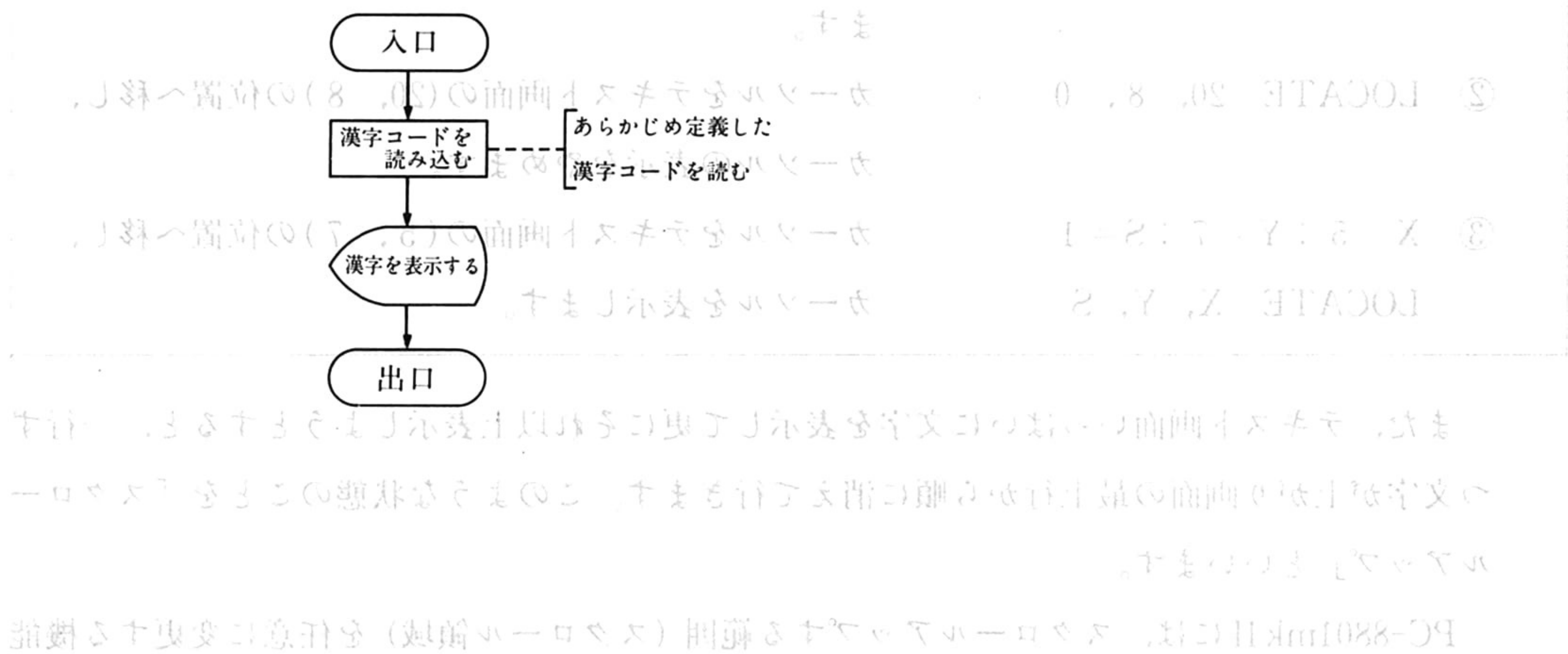
B

・サブルーチン

入口

漢字コードを読み込む　―　あらかじめ定義した漢字コードを読む

漢字を表示する

出口

（3）　プログラムに必要な命令の説明

利息比較一覧プログラムに新たに必要となる命令を説明します。

グラフィック画面上の表示位置を指定して漢字を表示したように，数字や文字をテキスト画面へ表示するときも表示位置を指定することができます。その方法は，カーソルを画面の任意の位置に移動させて，そこからPRINT命令を使って表示します。

カーソルをテキスト画面の任意の位置へ移動させるには，「LOCATE」命令を使います。

LOCATE命令の一般形式と書き方例は，次のとおりです。

(LOCATE命令の説明)

〈一般形式〉

LOCATE　列番号，行番号，カーソル表示スイッチ

- テキスト画面のカーソルの位置を指定し，カーソルをその位置へ移します。
- 列番号は，横の桁位置を指定します。画面サイズが40文字のときは0～39の範囲の値で，80文字のときは0～79の範囲の値で指定します。
- 行番号は，縦の行位置を指定します。画面サイズが20行のときは0～19の範囲の値で，25行のときは0～24の範囲の値で指定します。
- 列番号を省略すると，現在カーソルがセットされている桁とみなします。
- 行番号を省略すると，現在カーソルがセットされている行とみなします。
- カーソル表示スイッチには「0」か「1」を指定し，「0」を指定するとカーソルを表示しなくなり，「1」を指定するとカーソルを表示します。
- カーソル表示スイッチを省略すると，カーソルの表示は現在セットされている状態が継続されます。

〈書き方例〉

①	LOCATE　10，15	カーソルをテキスト画面の（10，15）の位置へ移します。
②	LOCATE　20，8，0	カーソルをテキスト画面の(20，8)の位置へ移し，カーソルの表示をやめます。
③	X＝5：Y＝7：S＝1 LOCATE　X，Y，S	カーソルをテキスト画面の(5，7)の位置へ移し，カーソルを表示します。

また，テキスト画面いっぱいに文字を表示して更にそれ以上表示しようとすると，一行ずつ文字が上がり画面の最上行から順に消えて行きます。このような状態のことを「スクロールアップ」といいます。

PC-8801mkⅡには，スクロールアップする範囲（スクロール領域）を任意に変更する機能があります。スクロール領域を変更するには「CONSOLE」命令を使います。

CONSOLE命令は「4.1.5」で説明しているので，ここではスクロール領域を変更する書き方を示します。

(CONSOLE命令の説明)

〈書き方例〉

①	CONSOLE　10，10	スクロール領域を第10行（11行目）から10行分にします。他の行はスクロールアップしない固定画面になります。
②	X＝5：Y＝18 CONSOLE　X，Y	スクロール領域を第5行（6行目）から18行分にします。

(4)　キーインするプログラム

(2)で示したフローチャート例に従ってコーディングしたプログラム例を次に示します。

【 プログラム例 】

```
10 ' リソク ヒカク イチラン プログラム
20 '
30 ' データ
40 DATA 4D78,4229,4866,3353,306C,4D77,2121,2121
50 DATA 3835,3662,2121,2121,2121,2121,2121,2121
60 DATA 472F,3865,2121,2121,2121,2121,2121,2121
70 DATA 2121,4576,2121,3A42,2121,4D42,2121,3662
80 DATA 2121,4961,2121,444C,2121,4D42,2121,3662
90 DATA 2121,444C,2121,434E,2121,4D42,2121,3662
100 DATA 347C,467C,3B58,446A,446A,347C,4D42,3662
```

（つづく）

```
110 ' *** ｼﾞｭﾝﾋﾞ ｼｮﾘ ***
120 CLS 3:SCREEN 0:WIDTH 80,20:CONSOLE 16,2,0
130 DIM A(8),B#(4):W1$=":"
140 ' ﾐﾀﾞｼ ﾋｮｳｼﾞ
150 N=160:C=2:Y=1:GOSUB *KAN.DIS
160 N=340:C=5:Y=25:GOSUB *KAN.DIS
170 N=474:C=5:Y=45:GOSUB *KAN.DIS
180 FOR Y=65 TO 125 STEP 20
190   N=0:C=4:GOSUB *KAN.DIS
200 NEXT Y
210 FOR Y=7 TO 13 STEP 2
220   LOCATE 21,Y:PRINT W1$
230 NEXT Y
240 ' *** ﾘｿｸ ｹｲｻﾝ ｼｮﾘ ***
250 CLS
260 INPUT "ｶﾞﾝｷﾝ = ";GANKIN#
270 IF GANKIN#=0 THEN CONSOLE 0,25,1:END
280 INPUT "ﾈﾝﾈﾝｺﾞ = ";NEN
290 LOCATE 50,3
300 PRINT USING "= ¥¥##,###,###";GANKIN#
310 LOCATE 55,5
320 PRINT USING "##";NEN
330 ' ｺﾞｳｹｲ ｹｲｻﾝ
340 FOR J=1 TO 4:B#(J)=GANKIN#:NEXT J
350 B#(1)=B#(1)*(1+0)^NEN
360 B#(2)=B#(2)*(1+.015)^NEN
370 B#(3)=B#(3)*(1+.0175)^NEN
380 B#(4)=B#(4)*(1+.055/2)^(2*NEN)
390 ' ﾒｲｻｲ ﾋｮｳｼﾞ
400 FOR Y=7 TO 13 STEP 2
410   W=(Y-5)/2
420   LOCATE 23,Y
430   PRINT USING "¥¥##,###,###";B#(W)
440 NEXT Y
450 GOTO 250
460 ' *** ｶﾝｼﾞ ﾋｮｳｼﾞ ｻﾌﾞﾙｰﾁﾝ ***
470 *KAN.DIS
480 FOR I=1 TO 8
490   READ X$:A(I)=VAL("&H"+X$)
500 NEXT I
510 FOR X=N TO N+140 STEP 20
520   W=(X-N)/20+1
530   PUT(X,Y),KANJI(A(W)),PSET,C,0
540 NEXT X
550 RETURN
```

【 プログラム例 】

・行番号110～450がメインルーチンで，行番号460～550が漢字表示用のサブルーチンです。

・行番号40～100のDATA文は，表示する漢字を漢字コードで表したものです。

・行番号120では，画面（テキスト画面とグラフィック画面の両方）の初期設定をします。

・行番号130のDIM文で定義する配列は，漢字の表示と四つの預金の元利合計の計算用に使います。

・行番号150～170および行番号190では，漢字表示に必要なパラメータをセットして漢字表示サブルーチンへ分岐します。

・行番号180～200のFOR～NEXT命令は，変数YをY座標の値として使うため初期値を「65」，最終値を「125」，増分を「20」にします。

・行番号210～230では，画面の指定した位置（21，Y）に「：」を表示します。

・行番号250のCLS命令は，スクロール領域内のみをクリアし，スクロール開始行の先頭にカーソルを移します。

・行番号260と280のINPUT命令によるデータの入力は，スクロール領域内で行われます。

・行番号270では，元金の入力で ⏎ キーだけが押されたときに，スクロール領域を初期値に戻して処理を終了します。

・行番号340では，四種類の預金の元利合計を求めるために，入力した元金を初期値として配列B＃の四つの配列要素にそれぞれ代入します。

・行番号350～380では，n年後のそれぞれの元利合計を配列B＃へ求めます。

・行番号400～440では，算出した元利合計を画面の指定したそれぞれの位置（23，Y）に表示します。

・行番号410では，変数Yが「7，9，11，13」と変化するのを「1，2，3，4」と変化するように計算します。

・行番号480～500では，DATA文の漢字コードを配列Aに読み込みます。

・行番号490の「A（I）＝VAL（"&H"＋X$）」は，読み込んだ漢字コードの先頭に「&H」を付けて16進数にした後配列Aへ代入します。

・行番号510～540では，グラフィック画面に漢字を表示します。

・行番号520は，行番号410の処理と同様に，変数Xの変化を「1，2，3……」と変化するように計算します。

・行番号530では，グラフィック画面の指定した位置（X，Y）に配列A(w）の漢字コードを指定された色（c）で表示します。

（5） プログラムのキーイン

設問 68 メモリーをクリアした後，例で示したプログラムをメモリーにキーインして下さい。

設問 69　正しくキーインされているかどうか，プログラムリストをすべてプリンタに印字して下さい。

【 プリンタ印字例 】

```
10 ' ﾘｿｸ ﾋｶｸ ｲﾁﾗﾝ ﾌﾟﾛｸﾞﾗﾑ
20 '
30 ' ﾃﾞｰﾀ
40 DATA 4D78,4229,4866,3353,306C,4D77,2121,2121
50 DATA 3835,3662,2121,2121,2121,2121,2121,2121
60 DATA 472F,3865,2121,2121,2121,2121,2121,2121
70 DATA 2121,4576,2121,3A42,2121,4D42,2121,3662
80 DATA 2121,4961,2121,444C,2121,4D42,2121,3662
90 DATA 2121,444C,2121,434E,2121,4D42,2121,3662
100 DATA 347C,467C,3B58,446A,446A,347C,4D42,3662
110 ' *** ｼﾞｭﾝﾋﾞ ｼｮﾘ ***
120 CLS 3:SCREEN 0:WIDTH 80,20:CONSOLE 16,2,0
130 DIM A(8),B#(4):W1$=":"
140 ' ﾐﾀﾞｼ ﾋｮｳｼﾞ
150 N=160:C=2:Y=1:GOSUB *KAN.DIS
160 N=340:C=5:Y=25:GOSUB *KAN.DIS
170 N=474:C=5:Y=45:GOSUB *KAN.DIS
180 FOR Y=65 TO 125 STEP 20
190   N=0:C=4:GOSUB *KAN.DIS
200 NEXT Y
210 FOR Y=7 TO 13 STEP 2
220   LOCATE 21,Y:PRINT W1$
230 NEXT Y
240 ' *** ﾘｿｸ ｹｲｻﾝ ｼｮﾘ ***
250 CLS
260 INPUT "ｶﾞﾝｷﾝ = ";GANKIN#
270 IF GANKIN#=0 THEN CONSOLE 0,25,1:END
280 INPUT "ﾈﾝｽｳ = ";NEN
290 LOCATE 50,3
300 PRINT USING "= ¥¥##,###,###";GANKIN#
310 LOCATE 55,5
320 PRINT USING "##";NEN
330 ' ｺﾞｳｹｲ ｹｲｻﾝ
340 FOR J=1 TO 4:B#(J)=GANKIN#:NEXT J
350 B#(1)=B#(1)*(1+0)^NEN
360 B#(2)=B#(2)*(1+.015)^NEN
370 B#(3)=B#(3)*(1+.0175)^NEN
380 B#(4)=B#(4)*(1+.055/2)^(2*NEN)
390 ' ﾒｲｻｲ ﾋｮｳｼﾞ
400 FOR Y=7 TO 13 STEP 2
410   W=(Y-5)/2
420   LOCATE 23,Y
430   PRINT USING "¥¥##,###,###";B#(W)
440 NEXT Y
450 GOTO 250
```

（つづく）

```
460 ' *** カンジ" ヒョウジ" サブ"ルーチン ***
470 *KAN.DIS
480 FOR I=1 TO 8
490   READ X$:A(I)=VAL("&H"+X$)
500 NEXT I
510 FOR X=N TO N+140 STEP 20
520   W=(X-N)/20+1
530   PUT(X,Y),KANJI(A(W)),PSET,C,0
540 NEXT X
550 RETURN
```

（6） 作成したプログラムの実行

設問 70　今メモリーにキーインしたプログラムを一度実行して下さい。

【 実行結果例 】

```
            利息比較一覧
                         元金  =    ¥500,000
                                 3  年後
  当 座 預 金 :     ¥500,000
  普 通 預 金 :     ¥522,839
  通 知 預 金 :     ¥526,712
 期日指定定期預金 :     ¥588,384

ガ"ンキン = ? ■
```

【 実行結果例の説明 】

・これは，元金に「500000」を，「3」を入力したときの例です。

・プログラムを実行すると，画面をクリアした後漢字を表示し，元金の入力待ちの状態になります。

・「利息比較一覧」は赤色，「元金」と「年後」は水色，「当座預金」，「普通預金」，「通知預金」，および「期日指定定期預金」は緑色で表示します。

・元金と年数を入力すると，まず入力した元金と年数を指定の位置に表示します。それから，四種類の預金の元利合計を算出し，指定した位置に表示します。

・元金と四つの元利合計は，それぞれ「億」の単位まで表示できます。

・年数は，「99」まで表示できます。

・元金，年数および四つの元利合計を表示すると，再び元金の入力待ちになります。

・処理を終了するには，元金の入力時に [↵] キーだけを押すか「0」を入力します。

・なお，漢字を消すには「CLS 2 [↵] 」を実行します。

設問 71　正しく実行できていたら，プログラムをドライブ2のフロッピーディスクへ「KANJI.1」という名前でセーブして下さい。

【実行結果例】

```
save "2:KANJI.1"
Ok
```

8.3 漢字の印字

漢字プリンタ(PC-8821/8822)に漢字を印字するには，それぞれの文字に付けられている漢字コードをプリンタへ出力します。そうすることによって，漢字を使って通信文書や報告書などを作ることができます。(ただし，PC-8821の場合は，漢字ROMと呼ばれるJIS第1水準の漢字コードを記録したプリンタ用のメモリーを追加しなければなりません。)

そこで，この項ではプリンタに漢字を印字する方法について学習します。

8.3.1 漢字をプリンタに印字するには

漢字を漢字プリンタに印字するには，「LPRINT」命令を使って初めにプリンタを漢字モードにします。その後で「LPRINT」命令を使って漢字コードを出力します。

さらに，漢字モードを解除するときも「LPRINT」命令を使います。

(LPRINT命令の説明)

〈一般形式1〉

LPRINT　CHR$(&H1B)；"K"；

・漢字プリンタを漢字が印字できる状態（漢字モード）にします。

・「K」は必ず大文字で指定します。

・この命令を実行した後の印字データはすべて漢字コードとして扱います。

・16進数の「&H1B」を10進数で「27」と指定してもかまいません。

〈一般形式2〉

LPRINT　CHR$(&H1B)；"H"；

・漢字プリンタの漢字を印字できる状態（漢字モード）を解除します。

・「H」は必ず大文字で指定します。

・この命令を実行した後の印字データはすべて普通の文字（英・数・カナ）として扱います。

〈一般形式3〉

LPRINT　CHR$(式1)；CHR$(式2)

漢字モードのとき，式1と式2で指定された漢字プリンタに印字します。

・式1と式2には，16進数で表す漢字コード（&Hxxyy）のうち，「&Hxx」を式1へ，「&Hyy」を式2へ指定します。

〈書き方例〉

① LPRINT　CHR$(&H1B)；"K"；

LPRINT　CHR$(&H33)；CHR$(&H58)；CHR$(&H3D)；CHR$(&H2C)

プリンタを漢字モードに切り替えて，「学習」という漢字を印字します。最初の「&H33」と次の「&H58」で「学」の文字を，三番目の「&H3D」と四番目の「&H2C」で「習」の文字を表します。

② LPRINT　CHR$(27)；"K"；

LPRINT　CHR$(&H33)；CHR$(&H58)；CHR$(&H3D)；CHR$(&H2C)

①と同じ結果になります。

③ LPRINT　CHR$(&H1B)；"H"；"テスト"

漢字モードを解除して「テスト」という普通の文字を印字します。

8.3.2 プログラム例(7)————漢字を印字するプログラム

ここで作成するプログラムの条件を次のように設定します。

【 作成の条件 】

・漢字は40文字まで印字できるようにします。

・漢字コードは16進コードで入力します。

・漢字コードの入力で ⏎ キーだけを押すと，そこまで入力した漢字を印字します。

・漢字を印字するとに，入力した漢字コードを上位2桁と下位2桁の16進数に分けます。

また，上の条件を満足するようなプログラムのフローチャートを作成すると，次のようになります。

【 フローチャート例 】

開始

プリンタを漢字モードにする

カウンタを初期値クリア

漢字コードの入力

漢字コードはヌルか？

YES

NO

カウンタに1を加算

カウンタは40を超えたか？

NO

YES

カウンタを初期値クリア

漢字コードを2つの16進数に分ける

漢字を印字

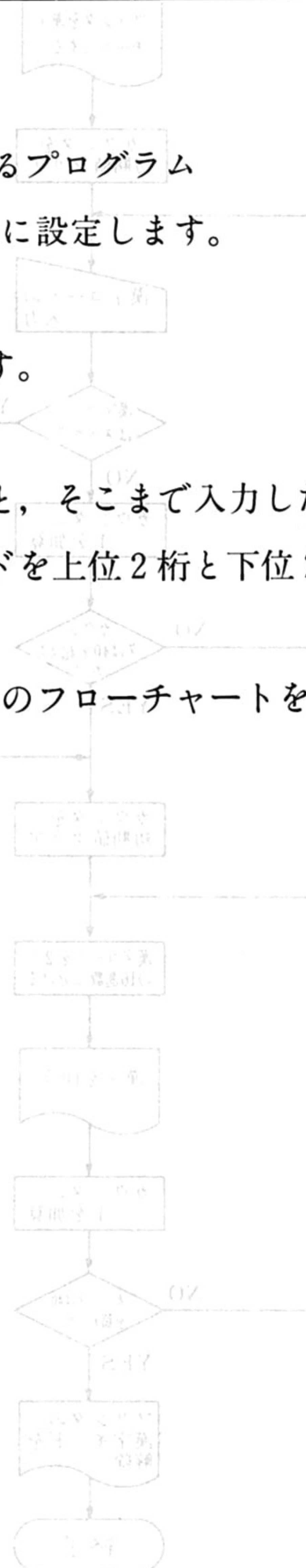
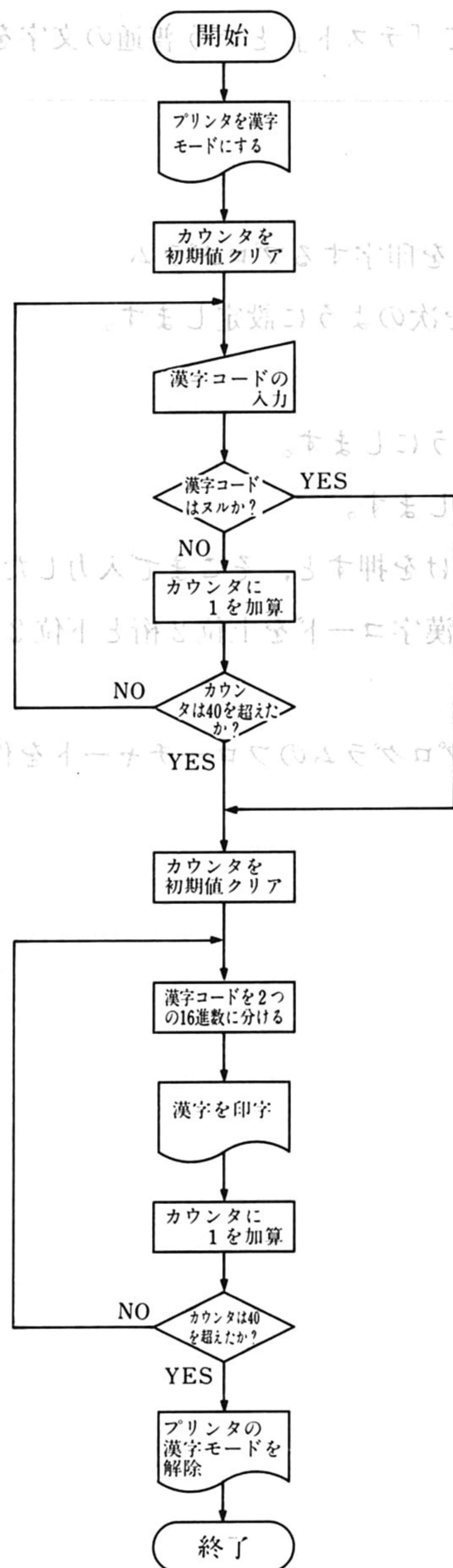
カウンタに1を加算

カウンタは40を超えたか？

NO

YES

プリンタの漢字モードを解除

終了

設問 72　例で示したフローチャートに従って，プログラムを作成して下さい。

【 プログラム例 】

```
10 R=40:DIM A$(40)
20 LPRINT CHR$(&H1B);"K";
30 FOR J=1 TO R
40   INPUT "ｶﾝｼﾞｺｰﾄﾞ xxxx = ";A$(J)
50   IF A$(J)="" THEN 70
60 NEXT J
70 FOR J=1 TO R
80   X$="&H"+LEFT$(A$(J),2):X=VAL(X$)
90   Y$="&H"+RIGHT$(A$(J),2):Y=VAL(Y$)
100  LPRINT CHR$(X);CHR$(Y);
110 NEXT J
120 LPRINT CHR$(&H1B);"H"
130 END
```

【 プログラム例の説明 】

・行番号10では，１行に印字する漢字の最大数を「40文字」とします。

・行番号20では，漢字プリンタを漢字モードにします。

・行番号40では，16進数の漢字コードを４桁で入力します。

・行番号50では，漢字コードの入力時に ⏎ キーだけが押され配列要素A$(J)にヌルストリングが入力されたとき，行番号70へ分岐します。

・行番号80では，４桁で入力した漢字コードの上位２桁を取り出し，数値に変換しています。

・行番号90では，４桁で入力した漢字コードの下位２桁を取り出し，数値に変換しています。

・行番号100では，漢字プリンタに漢字コードを出力して，漢字を印字しています。

・行番号120では，プリンタの漢字モードを解除して通常の印字モード(ハイデンシティパイカ)に戻しています。

設問 73　メモリーをクリアした後，例で示したプログラムをメモリーにキーインして下さい。

設問 74　正しくキーインされているかどうか，プログラムリストをすべて画面に表示して確認して下さい。

【 プログラムリスト表示例 】

```
list
10 R=40:DIM A$(40)
20 LPRINT CHR$(&H1B);"K";
30 FOR J=1 TO R
40   INPUT "ｶﾝｼﾞｺｰﾄﾞ xxxx = ";A$(J)
50   IF A$(J)="" THEN 70
60 NEXT J
70 FOR J=1 TO R
80   X$="&H"+LEFT$(A$(J),2):X=VAL(X$)
90   Y$="&H"+RIGHT$(A$(J),2):Y=VAL(Y$)
100  LPRINT CHR$(X);CHR$(Y);
110 NEXT J
120 LPRINT CHR$(&H1B);"H"
130 END
Ok
```

設問 75　正しくキーインされていたら，次のデータでプログラムを実行して下さい。

（入力するデータ）

漢字	漢字コード
漢	3441
字	3B7A
印	3075
字	3B7A

漢字	漢字コード
の	244E
方	4A7D
法	4B21

【 実行結果例 】

（画面）

```
run
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? 3441
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? 3B7A
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? 3075
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? 3B7A
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? 244E
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? 4A7D
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? 4B21
ｶﾝｼﾞｺｰﾄﾞ xxxx = ? ←――――ここで ⏎ キーだけを押した。
Ok
```

漢字印字の方法

【 実行後の説明 】

・データを全部入力したら, ⏎ キーだけをキーインします。すると, 入力した漢字コードに対する漢字プリンタに印字されます。

・プリンタは, 漢字を印字した後漢字モードを解除しています。

設問 76　正しく実行できていたら, プログラムをドライブ2のフロッピーディスクへ「KANJI.2」という名前でセーブして下さい。

【 実行結果例 】

```
save "2:KANJI.2"
Ok
```

8.3.3 漢字を印字するときの注意点

漢字をプリンタに印字する場合, 漢字プリンタの機能や制限から注意しなければならない点がいくつかあります。

漢字を印字するときの注意点は, 次のとおりです。

【 漢字印字の注意点 】

・漢字１文字の大きさは通常の文字 (ハイデンシティパイカ) の1.5文字分に相当します。漢字と他のモードの文字を同時に使用する場合, 印字の桁指定に注意します。

・漢字モードの場合, １行に印字できる最大文字数は「53」です。

・漢字モードのとき, 漢字を拡大して印字する (拡大モード) ことは可能ですが, 縮小して印字することは不可能です。

・漢字にアンダーラインを付けて印字することは可能です。

漢字印字の方法

【実行後の説明】
・データを全部入力したら，↵キーだけをキーインします。すると，入力した漢字コードに対する漢字がプリンタに印字されます。
・プリンタは，漢字を印字した後漢字モードを解除しています。

設問 76　正しく実行できていたら，プログラムをドライブ2のフロッピーディスクへ「KANJI.2」という名前でセーブして下さい。

【実行結果例】

```
save "2:KANJI.2"
Ok
```

8.3.3　漢字を印字するときの注意点

漢字をプリンタに印字する場合，漢字プリンタの機能や制限から注意しなければならない点がいくつかあります。

漢字を印字するときの注意点は，次のとおりです。

【漢字印字の注意点】
・漢字1文字の大きさは通常の文字（ハイデンシティ（パイカ））の1.5文字分に相当します。漢字と他のモードの文字を同時に使用する場合，印字の桁指定に注意します。
・漢字モードの場合，1行に印字できる最大文字数は「53」です
・漢字モードのとき，漢字を拡大して印字する（拡大モード）ことは可能ですが，縮小して印字することは不可能です。
・漢字にアンダーラインを付けて印字することは可能です。

9章　ゲームプログラム入門

この章では，これまでに学習したいろいろな命令を使って画面上で動きのあるプログラムの作り方を説明します。

9.1　画面上の文字をプログラムで動かすには

テキスト画面上での左から右への動き，左右往復の動き，そして斜めの動きについて説明します。

9.1.1　左から右へ動かす

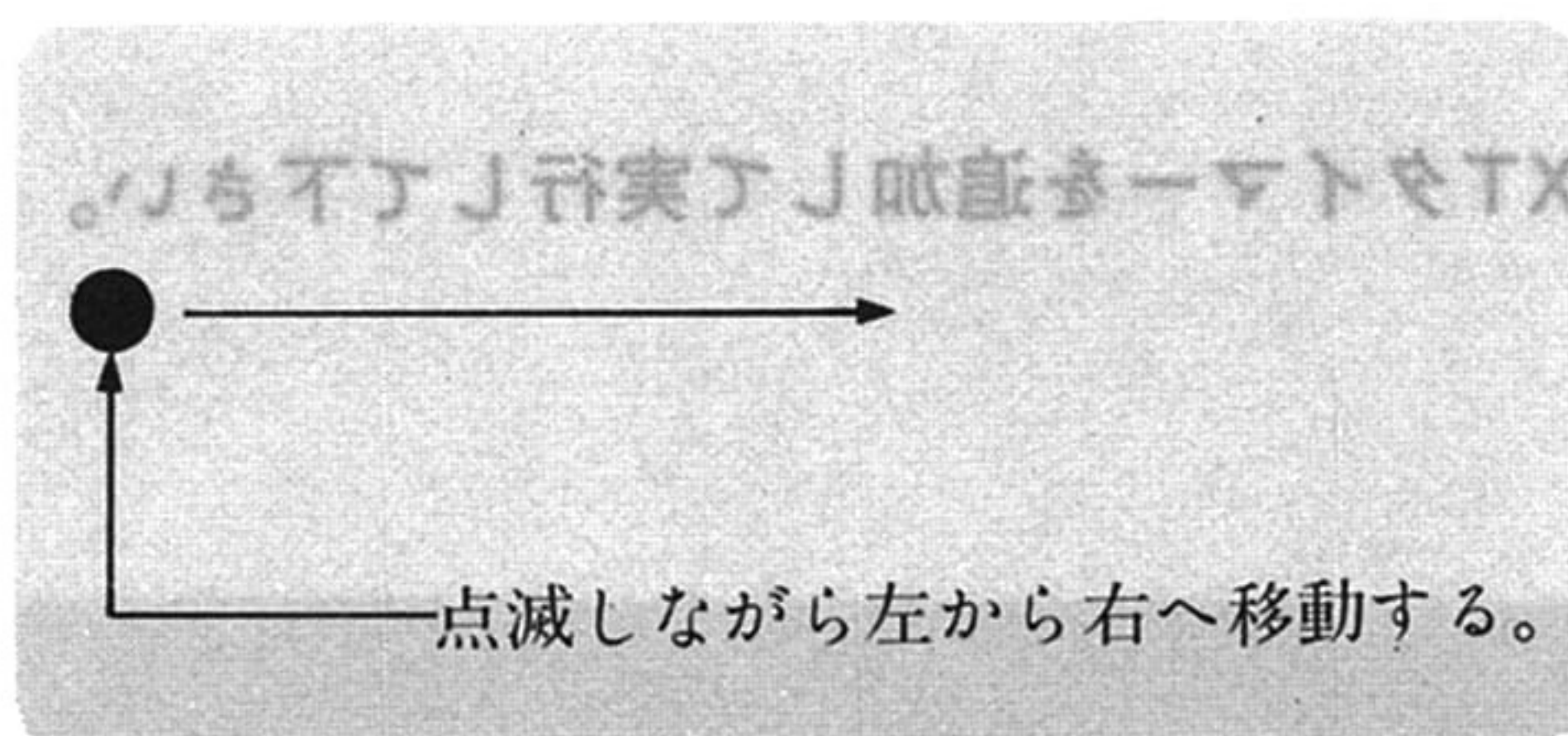
設問 77　メモリーをクリアした後，次のプログラムをキーインして実行して下さい。

【 キーインするプログラム 】

```
10 WIDTH 80,25:CONSOLE 0,25,0,1
20 CLS
30 FOR X=0 TO 79
40   LOCATE X,5:PRINT "●";
50   LOCATE X,5:PRINT " ";
60 NEXT X
70 END
```

【 実行結果例 】

●──────→

点滅しながら左から右へ移動する。

【 実行後の説明 】

・「●」のグラフィックシンボル文字を左から右へ動かすプログラムです。

・行番号40で「●」を表示し，行番号50でその「●」をスペースで消しています。

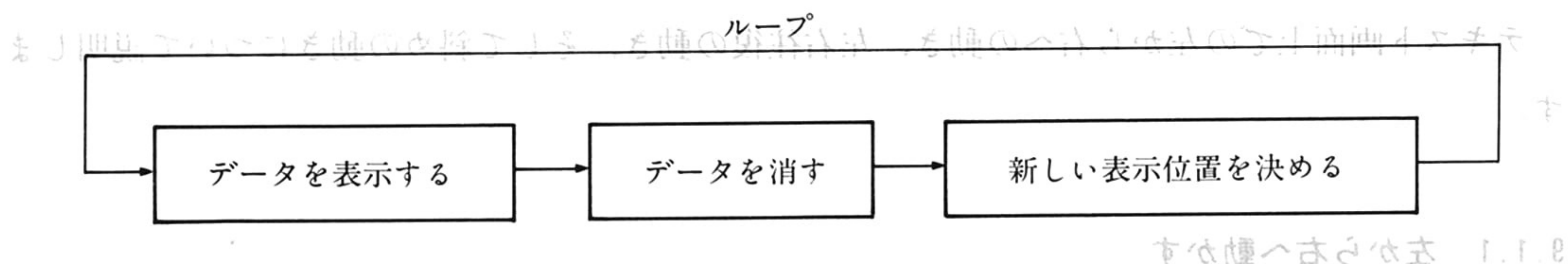
・行番号40～50の処理を行番号30～60で80回繰り返しています。

・行番号40と50のLOCATE命令で，「●」を表示，および消去する位置を左から右へ順に移動させています。そうすると，「●」が点滅しながら右へ動いているように見えます。

以上のことが，画面上の文字や絵を動かす基本的なテクニックです。

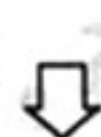
このような表示データを動かす原理を図に表すと，次のようになります。

ループ

データを表示する → データを消す → 新しい表示位置を決める

また，現在のプログラムでは左から右へ1桁ずつ表示文字が移動しますが，この移動速度をコントロールすることもできます。

・移動速度を2倍にする場合

行番号30を　　FOR　X＝0　TO　79

⇩

FOR　X＝0　TO　79　STEP　2　　に変更する。

・移動速度を遅くする場合

行番号40と50の間に　　FOR　I＝1　TO　25：NEXT　I　　を追加する。

このように何の処理もしないFOR～NEXT文は，時間をかせぐために使われます。このFOR～NEXT文を特に「FOR～NEXTタイマー」と呼び，「25」の値を大きくしたり小さくしたりすることによって時間を調整することができます。

設問 78　今メモリーにあるプログラムに，FOR～NEXTタイマーを追加して実行して下さい。

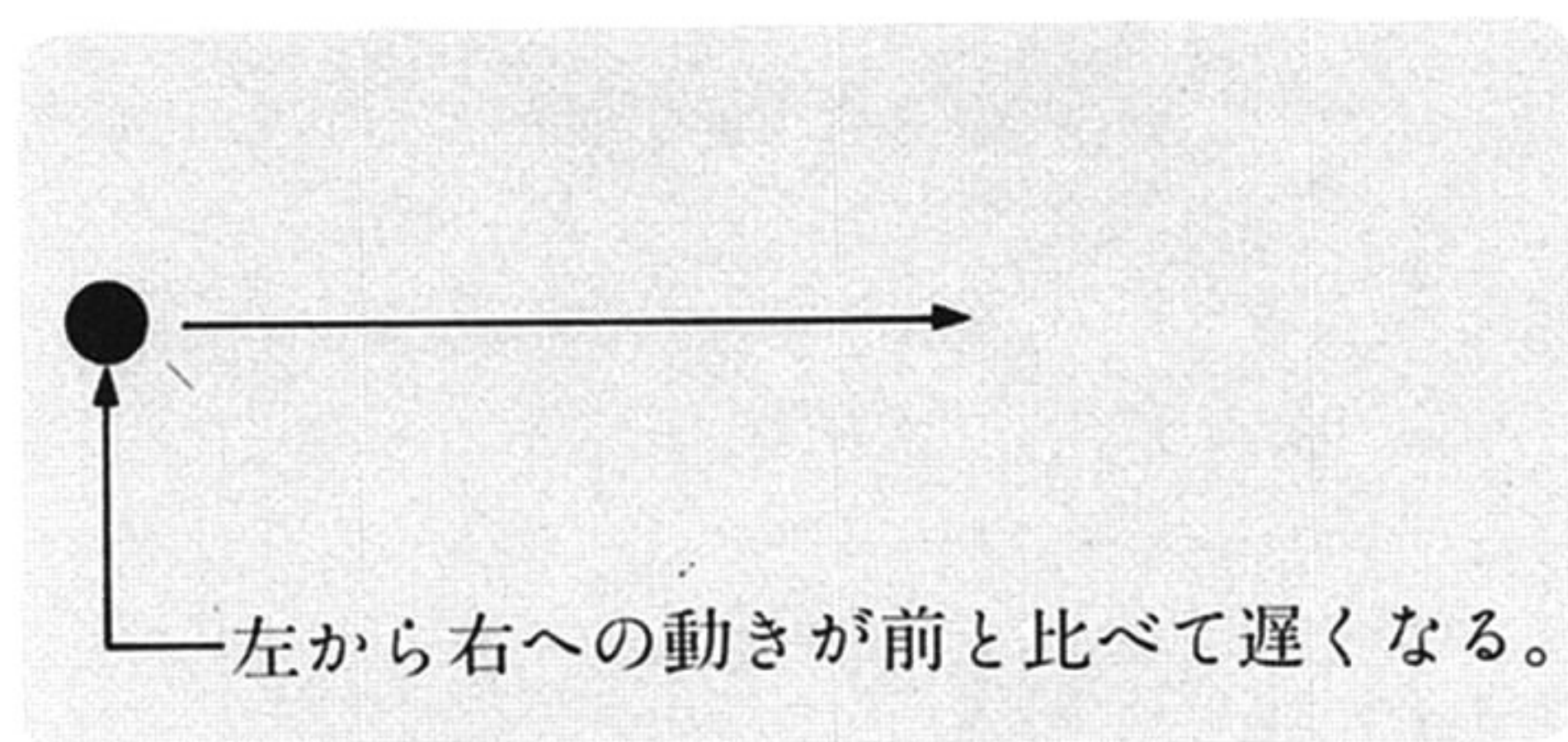
【 追加する命令 】

```
45   FOR I=1 TO 25:NEXT I
```

【 確認用プログラムリスト 】

```
10 WIDTH 80,25:CONSOLE 0,25,0,1
20 CLS
30 FOR X=0 TO 79
40   LOCATE X,5:PRINT "●";
45   FOR I=1 TO 25:NEXT I
50   LOCATE X,5:PRINT " ";
60 NEXT X
70 END
```

【 実行結果例 】

左から右への動きが前と比べて遅くなる。

【 実行後の説明 】

・行番号45にFOR～NEXTタイマーを追加することで,「●」の表示時間を長くしてから消すという動作をもっています。

・ここでのFOR～NEXTタイマーは, 移動速度をコントロールする目的とは別に, 動きをなめらかにするという目的にも使っています。

9.1.2 左右往復に動かす

設問 79 今メモリーに入っているプログラムを修正して「●」が左右往復の動きをするようにするため, 次の命令文を追加して下さい。

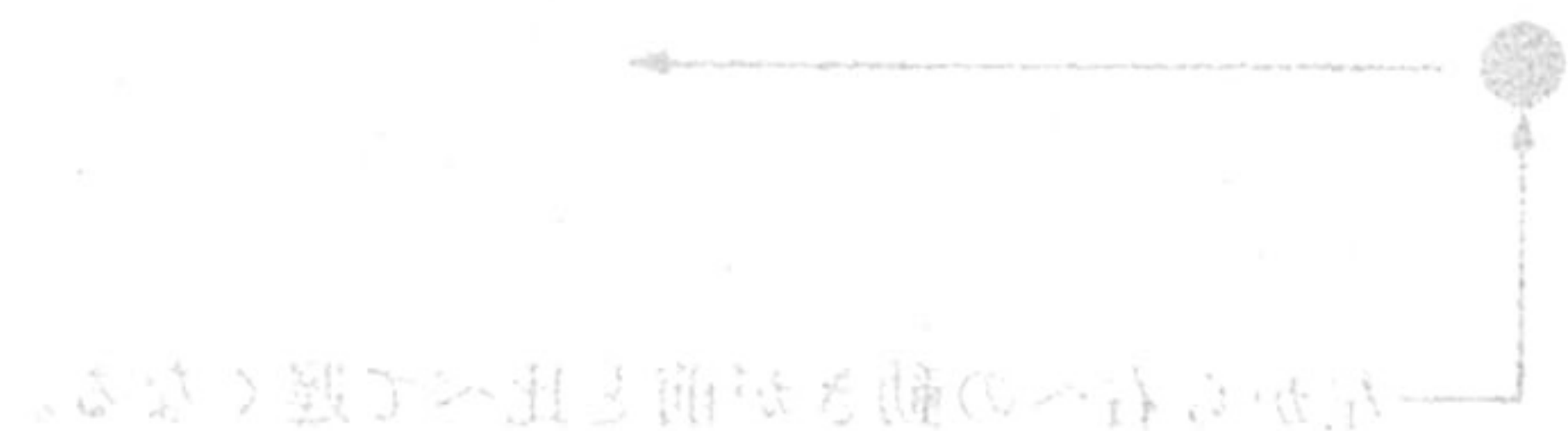
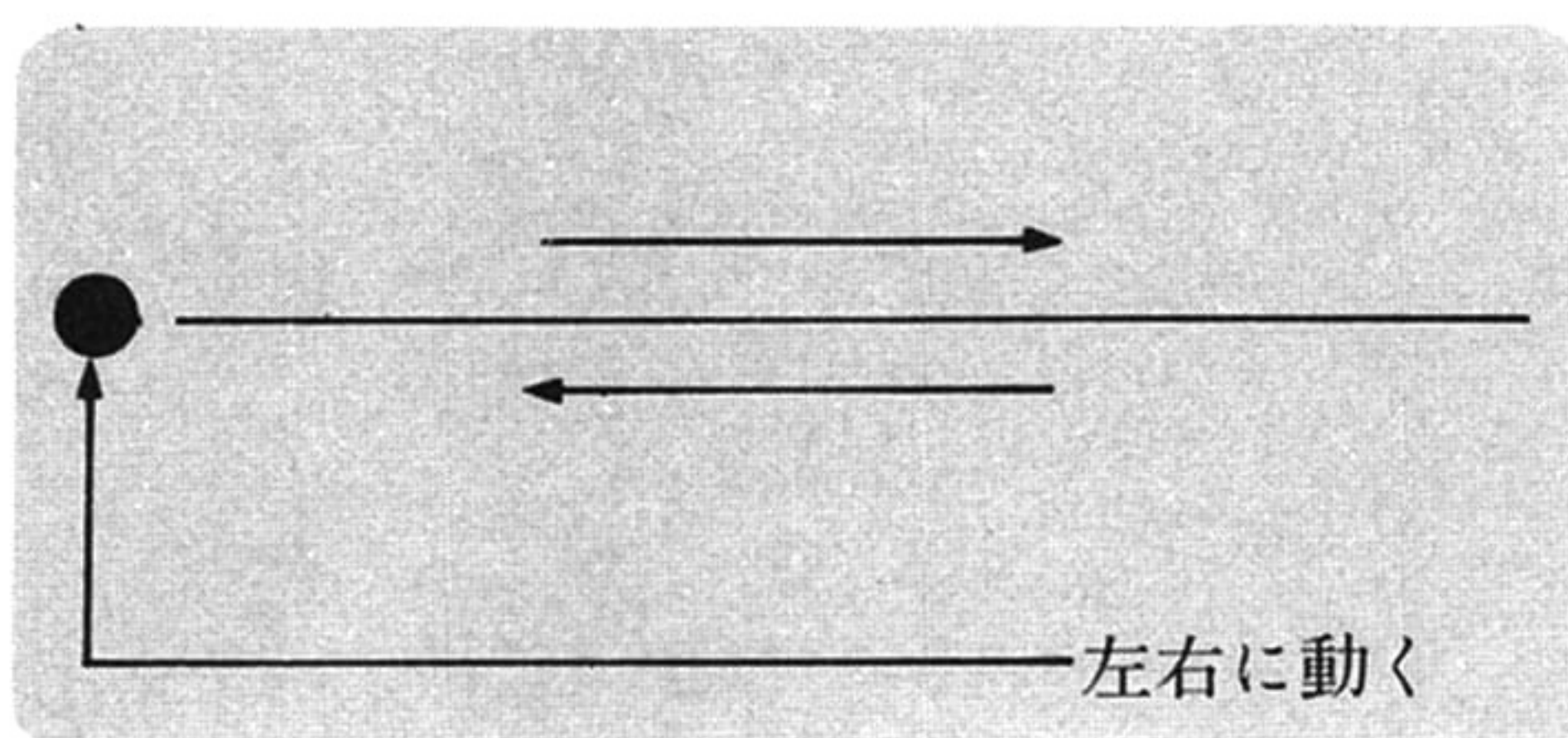
【 追加する命令 】

```
30 XW=1
55   X=X+XW:IF X>79 OR X<0 THEN XW=-XW:GOTO 55
60 GOTO 40
```

【 確認用プログラムリスト 】

```
10 WIDTH 80,25:CONSOLE 0,25,0,1
20 CLS
30 XW=1
40   LOCATE X,5:PRINT "●";
45   FOR I=1 TO 25:NEXT I
50   LOCATE X,5:PRINT " ";
55   X=X+XW:IF X>79 OR X<0 THEN XW=-XW:GOTO 55
60 GOTO 40
70 END
```

【 実行結果例 】

左右に動く

【 実行後の説明 】

・行番号40～60の処理を繰り返すことで，左右の往復の動きを行っています。

・行番号30の「XW＝１」は移動の速度を決めています。

・行番号55では,「●」の表示の桁位置を指定する変数Xの値を決めています。さらにIF文では,「●」が画面の右端または左端にきたときに移動する方向を変えています。

このように，表示する文字が指定した位置にきたとき移動の動きを逆にするというのが，はね返す処理の基本になります。

9.1.3 斜めに動かす

設問 79　今メモリーに記憶されているプログラムにＹ軸方向の要素を加えて斜めの動きをするようにするため，次の命令文を追加して下さい。

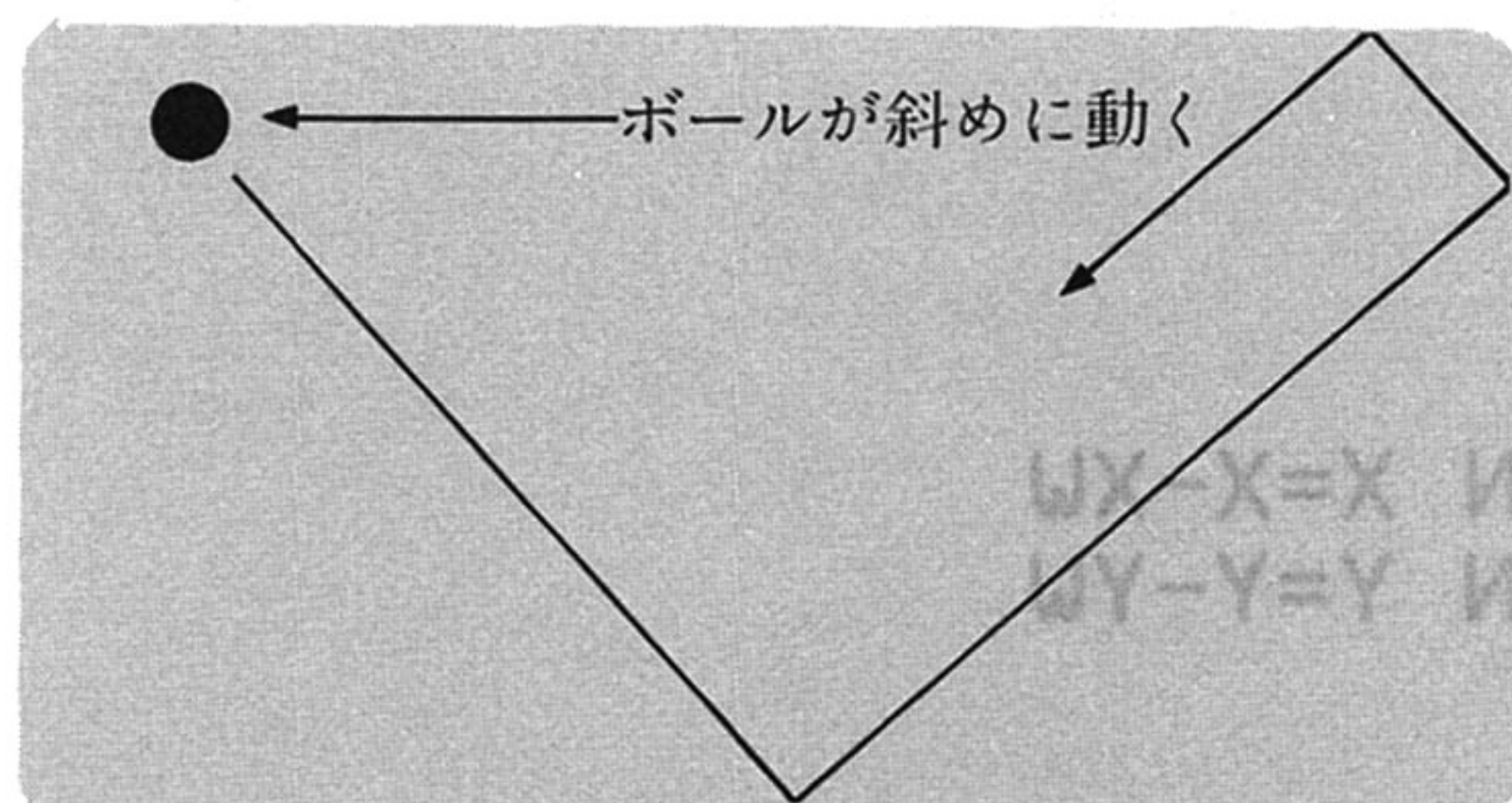
【 追加する命令 】

```
30 XW=2:YW=1
40   LOCATE X,Y:PRINT "●";
50   LOCATE X,Y:PRINT " ";
55   X=X+XW:IF X>79 OR X<0 THEN XW=-XW:GOTO 55
57   Y=Y+YW:IF Y>24 OR Y<0 THEN YW=-YW:GOTO 57
```

【 確認用プログラムリスト 】

```
10 WIDTH 80,25:CONSOLE 0,25,0,1
20 CLS
30 XW=2:YW=1
40   LOCATE X,Y:PRINT "●";
45   FOR I=1 TO 25:NEXT I
50   LOCATE X,Y:PRINT " ";
55   X=X+XW:IF X>79 OR X<0 THEN XW=-XW:GOTO 55
57   Y=Y+YW:IF Y>24 OR Y<0 THEN YW=-YW:GOTO 57
60 GOTO 40
70 END
```

【 実行結果例 】

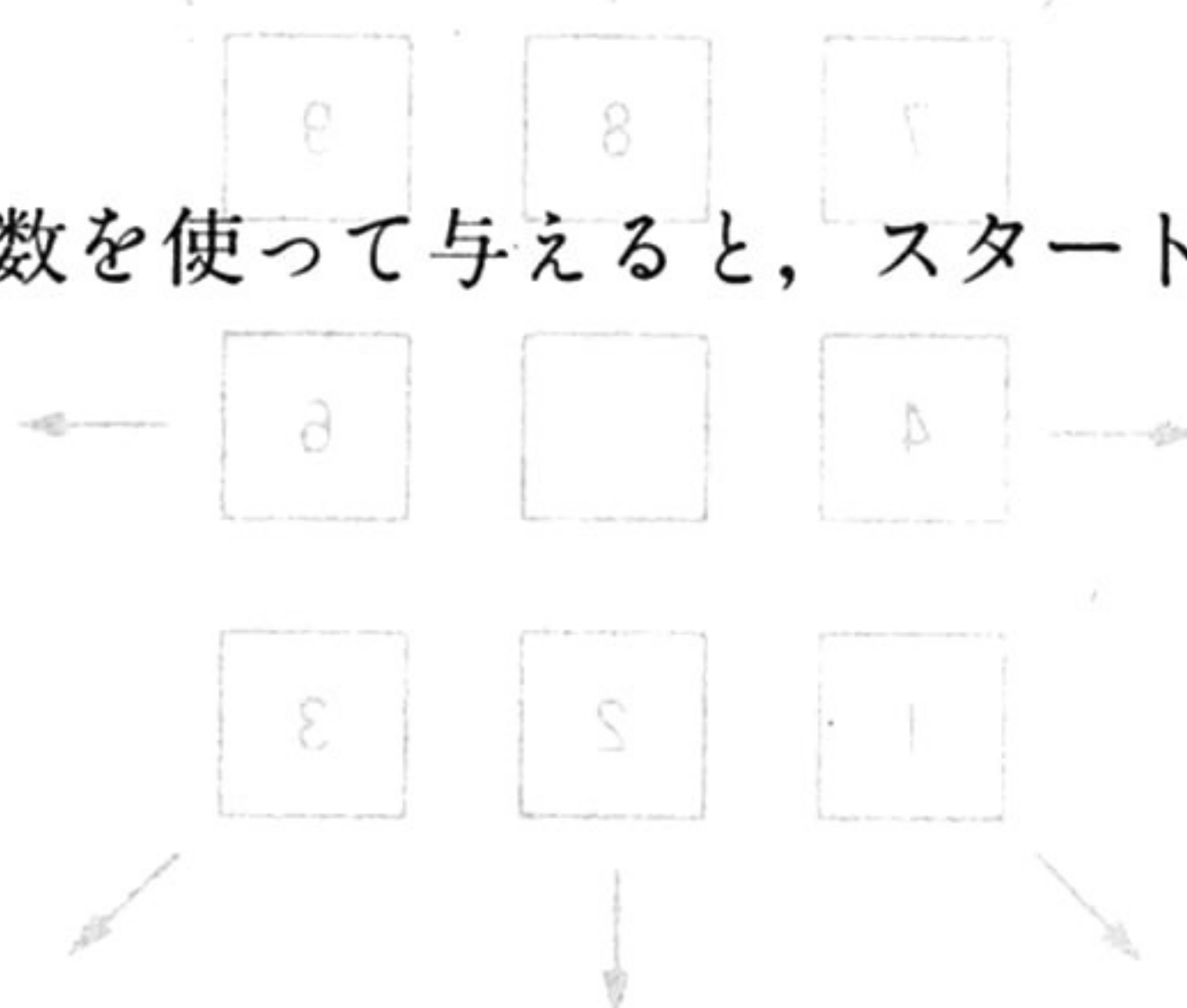
ボールが斜めに動く

【 実行後の説明 】

・前のプログラムにY軸方向の動きをX軸方向と同様に追加したプログラムです。

・行番号30で「XW＝２：YW＝1」としているので，X軸方向には２桁ずつ，Y軸方向には１桁ずつ動きます。

・動く速さは，行番号30のXWとYWの値を変更することでコントロールできます。

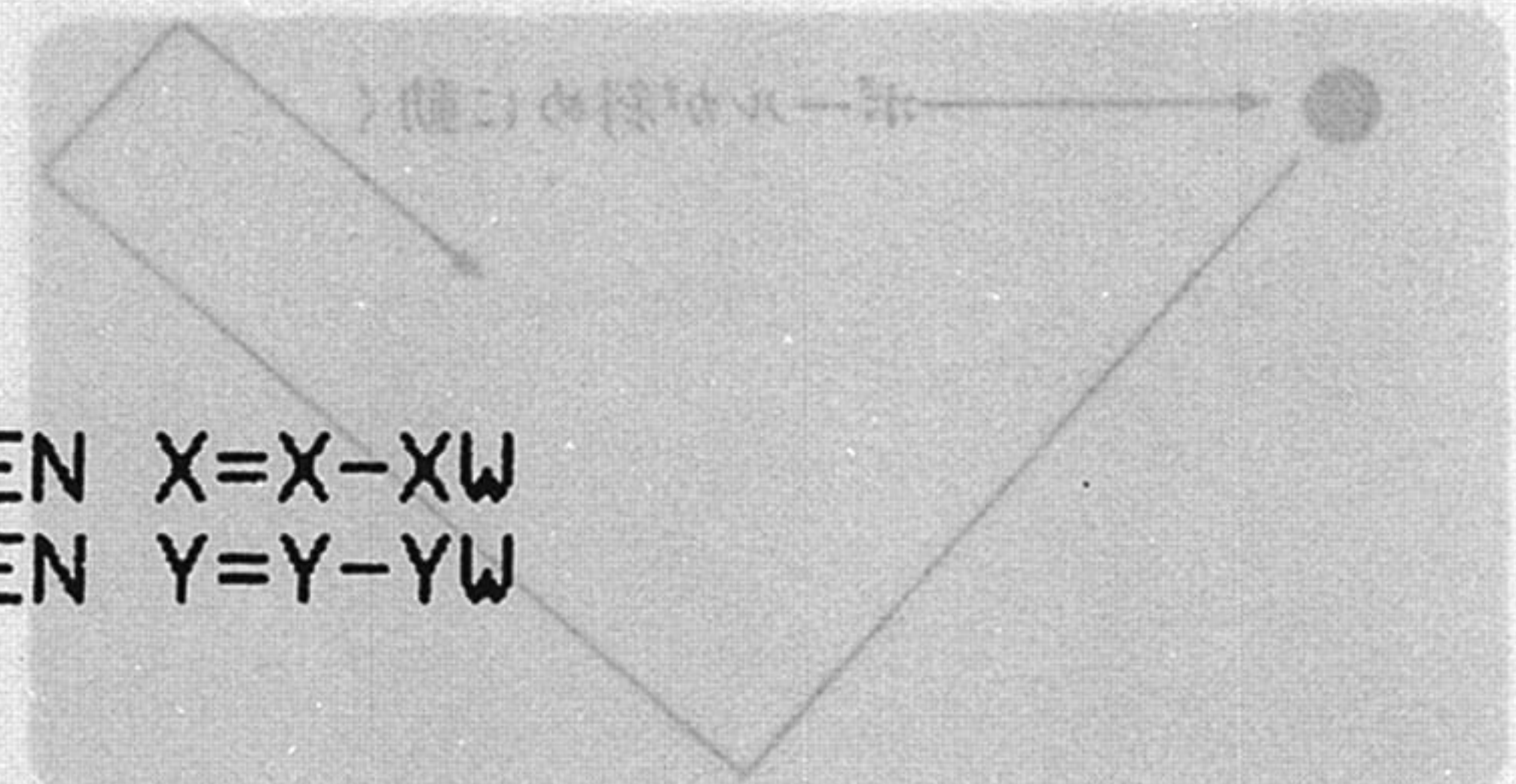
・動く範囲は，行番号55と57のIF文でチェックする範囲を変更することで，自由に設定できます。

・X，Y座標の初期値をRND関数を使って与えると，スタートする位置も自由に設定できるようになります。

9.2　画面上の文字をキー操作で動かすには

これまでのプログラムではプログラム内部で自動的に表示文字を動かしていましたが，ここではキー入力に応じて画面上のデータを動かす方法を説明します。

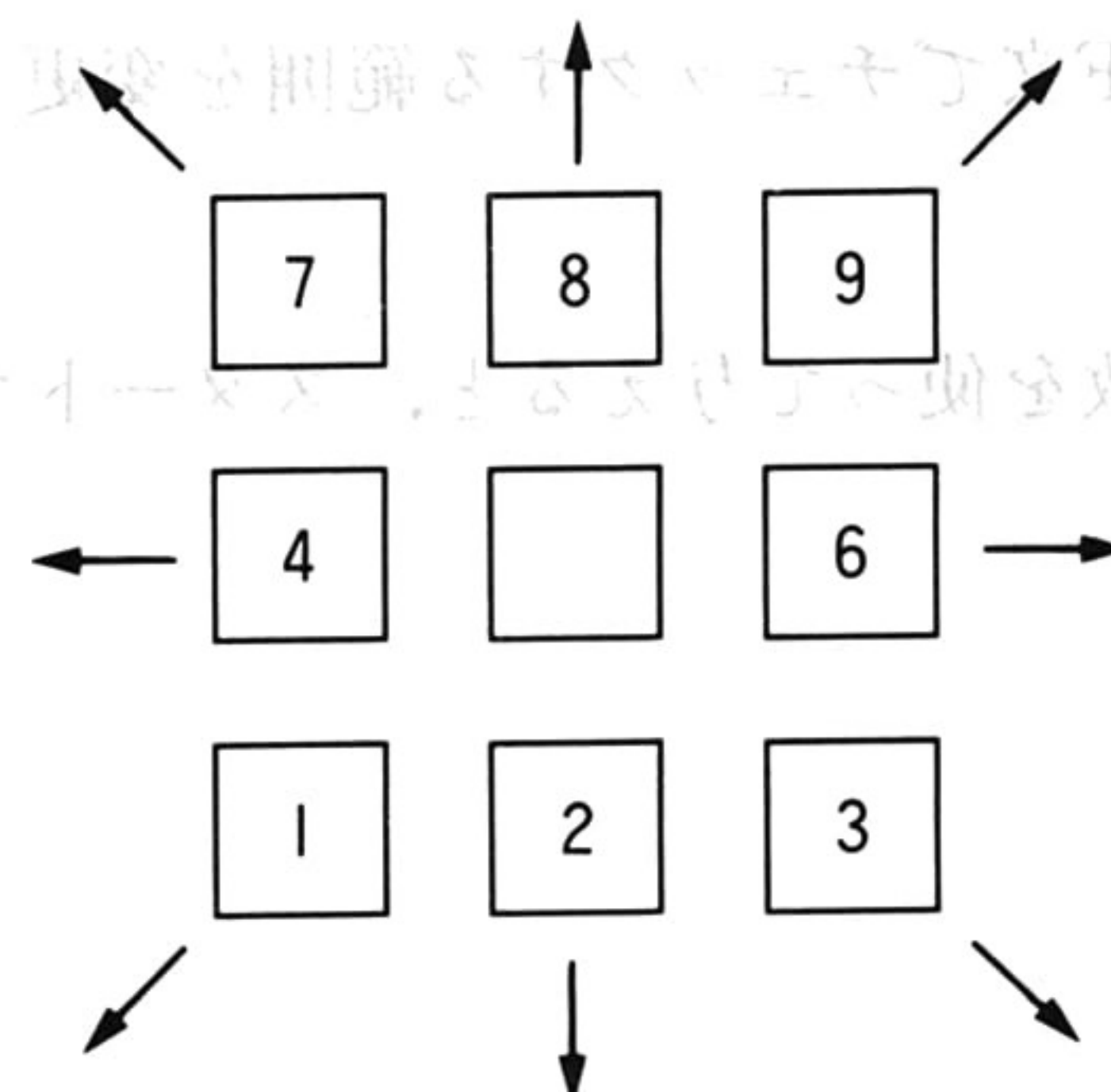
設問 80　メモリーをクリアしてから，次のプログラムをキーインして実行して下さい。

【 キーインするプログラム 】

```
10 WIDTH 80,25:CONSOLE 0,25,0,1
20 CLS
30 X=39:Y=11:LOCATE X,Y:PRINT "●";
40 ' --- LOOP ---
50 A$=INKEY$:IF A$="" THEN 50
60 IF A$<"1" OR A$>"9" OR A$="5" THEN 50
70 LOCATE X,Y:PRINT " ";
80 ON VAL(A$) GOTO 90,100,110,120,130,140,150,160,170
90   XW=-2:YW=1 :GOTO 200
100  XW=0 :YW=1 :GOTO 200
110  XW=2 :YW=1 :GOTO 200
120  XW=-2:YW=0 :GOTO 200
130  GOTO 50
140  XW=2 :YW=0 :GOTO 200
150  XW=-2:YW=-1:GOTO 200
160  XW=0 :YW=-1:GOTO 200
170  XW=2 :YW=-1:GOTO 200
200 X=X+XW:IF X>79 OR X<0 THEN X=X-XW
210 Y=Y+YW:IF Y>24 OR Y<0 THEN Y=Y-YW
220 LOCATE X,Y:PRINT "●";
230 GOTO 50
240 END
```

【 実行前の説明 】

・テンキーの「1」～「9」の数字キーを使って「●」を動かすプログラムです。

・行番号90～170のXW，YWに±2，±1または0をセットして，全部で八つの方向を次の図のようにコントロールしています。

↖	↑	↗
7	8	9
← 4		6 →
1	2	3
↙	↓	↘

・まず画面のほぼ中央に「●」を表示し，キー入力に応じて八つの方向へ移動します。

・行番号50～60で「５」を除く「１～９」のキー入力を受け，行番号80の「ON　VAL(A$)　GOTO ……」でキーの値に応じてXWとYWに移動数をセットします。

・行番号200では，Xの値が右端または左端を表す場合に，移動の方向を逆にします。

・行番号210では，行番号200と同様の処理をY軸について行います。

・行番号220で新しい表示位置に「●」を表示します。

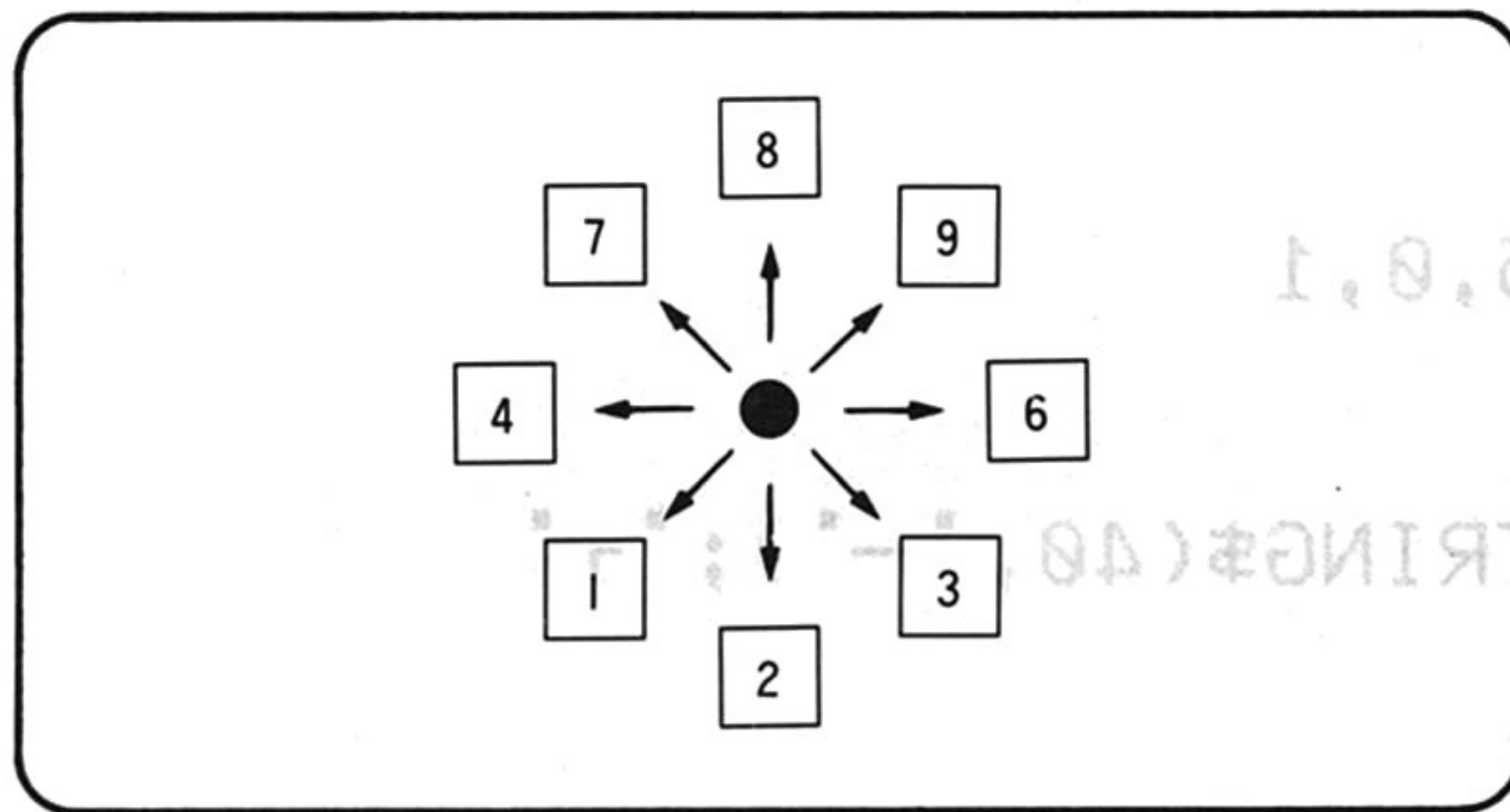
【 実行結果例 】

9.3　プログラム例(8)――――簡単なゲームプログラム

ここでは9.1，9.2でのテクニックを利用した簡単なゲームを紹介します。

このゲームはスカッシュゲームの一種で，飛んできたボールをラケットではね返すものです。5回ミスをするとゲームが終了します。

プログラムリストは次のとおりです。ただし，このプログラムをキーインする場合は，初めにメモリークリアを行って下さい。

【プログラムリスト】

```
100 WIDTH 80,25:CONSOLE 0,25,0,1
110 CLS
120 COLOR 4
130 LOCATE 19,1:PRINT "┌";STRING$(40,"─");"┐"
140 FOR I=2 TO 20
150   LOCATE 19,I:PRINT "|";
160   LOCATE 60,I:PRINT "|";
170 NEXT I
180 COLOR 7
190 AC=0
200 T=100:GOSUB 610
210 LOCATE 1,23:INPUT "ﾘﾀｰﾝｷｰ ｦ ｵｼﾃ ｸﾀﾞｻｲ       ",S$
220 LOCATE 1,23:PRINT SPACE$(20)
230 LOCATE 0,0:PRINT "ﾉｺﾘ ﾉ ﾎﾞｰﾙ";5-AC:CX=37
240 X=INT(RND(1)*38)+20
250 LOCATE CX,20:PRINT "_____";
260 FOR I=1 TO 1500:NEXT I
270 T=40:GOSUB 610
280 Y=2:XW=1:YW=1
290 LOCATE X,Y:PRINT " ";
300 X=X+XW
310 IF X<=20 OR X>=59 THEN XW=-XW:T=10:GOSUB 610
320 Y=Y+YW
330 IF X>CX-1 AND X<CX+5 AND Y=20 THEN SW=1 ELSE SW=0
340 IF Y=2 OR SW=1 OR Y=24 THEN YW=-YW:T=10:GOSUB 610
350 IF Y<24 THEN 370
360 AC=AC+1:LOCATE CX,20:PRINT SPACE$(7);:GOTO 230
370 IF AC<5 THEN 400
380 LOCATE X,Y:PRINT " ";
390 LOCATE CX,20:PRINT SPACE$(5);:GOTO 190
400 COLOR AC+1:LOCATE X,Y:PRINT "●";:COLOR 7
410 GOSUB 510:GOSUB 510
420 GOTO 290
```

（つづく）

```
500 ' --- ラケット イドウ ---
510 A$=INKEY$
520 IF A$><"4" AND A$><"6" THEN 580
530 IF A$="4" THEN CW=-4:GOTO 550
540 IF A$="6" THEN CW=4
550 CX=CX+CW:LOCATE CX-CW,20:PRINT SPACE$(5);
560 IF CX<20 THEN CX=20:GOTO 580
570 IF CX>55 THEN CX=55
580 LOCATE CX,20:PRINT "_____";
590 RETURN
600 ' --- ブザー ---
610 BEEP 1:FOR J=1 TO T:NEXT J:BEEP 0:RETURN
```

【 実行手順 】

① 「RUN ⏎」とキーインしてプログラムを開始させます。

② 画面上にコートが表示されます。

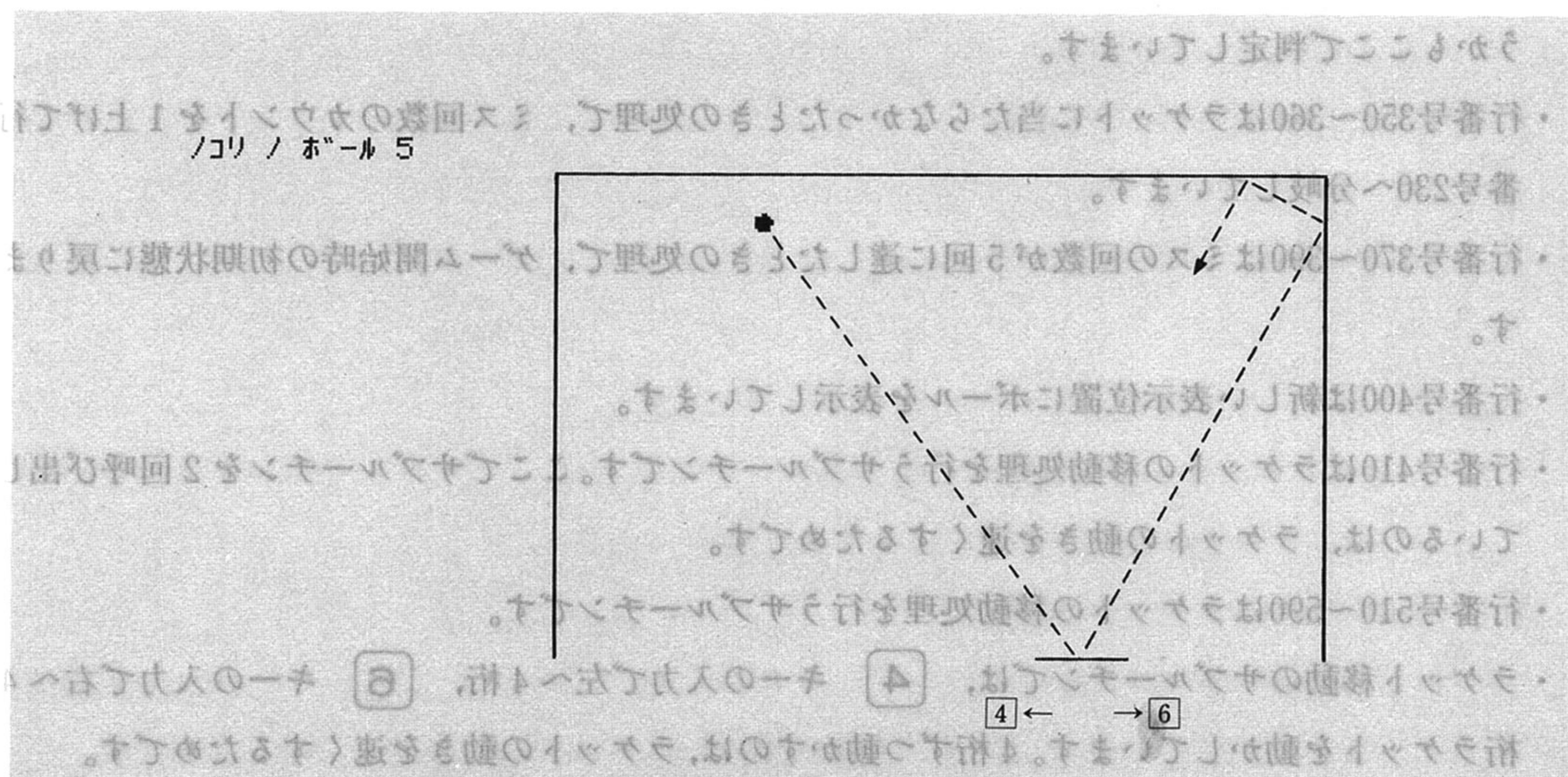
③ 続いて「リターンキー ヲ オシテ クダサイ」のメッセージが表示されるので ⏎ キーを押します。するとゲームが始まります。

④ コートの上端からボールが飛んでくるので，ラケットを 4 または 6 のキーで左右に動かしてボールをはね返します。

⑤ ミスが5回でゲームは終了し，再び③に戻ります。

【 実行結果例 】

ノコリ ノ ボール 5

4← →6

【 プログラムの説明 】

・画面をクリアした後，行番号120～170でコートを描きます。

・行番号190の「AC＝0」はミスの回数をカウントするカウンタで，初期値を0としています。

・行番号230は残りのボール数を表示します。ゲーム開始はAC＝0で，ボール数は「5」になります。

・行番号240は最初のボールの位置のX座標を，RND関数を使ってコートの範囲の20～57の間で設定しています。その考え方を次に示します。

初めに「RND(1)」で，たとえば「.123456」のような乱数を発生させます。

次に，発生した乱数を「RND(1)*38」として38倍すると，「4.69133」のように1桁以上の整数部ができます。

更に，小数部は不要なので，「INT」関数を使って「INT (RND(1)*38」のようにして整数「4」を求めます。

しかし，コート範囲は20桁目～59桁目なので，求めた値が20以上でなければなりません。そこで，「INT (RND(1)*38)＋20」とすればよいことになります。

このように，RND関数はゲームを作るときは必ずといっていいほど出てきます。

・行番号250はラケットをコートの下端の中央に表示します。

・行番号260はボールが出てくるまでに間をとるFOR～NEXTタイマーです。

・行番号280はボールが移動するときのX座標とY座標のきざみ値を初期設定します。

・行番号290～420が繰り返し処理の範囲になります。この繰り返し処理の中でボールの移動，ラケットでのはね返し処理を行います。

・行番号300～310はボール位置をX座標について判定しています。

・行番号320～340はボール位置をY座標について判定しています。また，ラケットに当たったかどうかもここで判定しています。

・行番号350～360はラケットに当たらなかったときの処理で，ミス回数のカウントを1上げて行番号230へ分岐しています。

・行番号370～390はミスの回数が5回に達したときの処理で，ゲーム開始時の初期状態に戻ります。

・行番号400は新しい表示位置にボールを表示しています。

・行番号410はラケットの移動処理を行うサブルーチンです。ここでサブルーチンを2回呼び出しているのは，ラケットの動きを速くするためです。

・行番号510～590はラケットの移動処理を行うサブルーチンです。

・ラケット移動のサブルーチンでは，［4］キーの入力で左へ4桁，［6］キーの入力で右へ4桁ラケットを動かしています。4桁ずつ動かすのは，ラケットの動きを速くするためです。

・行番号560はラケットが左端にきたとき，行番号570はラケットが右端にきたときの処理で，ラケットがコートをはずれないようにしています。

・行番号610はブザーを鳴らすためのサブルーチンです。変数Tに引き渡す値によって，ブザーの鳴る長さを調節できます。

以上でプログラムの説明は終わりますが，プログラムの内容が十分に把握できたら，はね返したときの点数カウントと点数表示などの機能を追加して，よりおもしろいゲームに改造してみて下さい。

また，このプログラムで新たに「COLOR」命令と「BEEP」命令を使っています。これらの命令の説明を次に示します。

(COLOR 命 令 の 説 明)

〈一般形式〉

COLOR　ファンクションコード，バックグラウンドカラー，ボーダーカラー，フォアグラウンドカラー

・画面の各部の色を指定します。

・ファンクションコードは，表示しようとするテキスト画面の文字に次の機能を与えます。

(白黒モードの場合／CONSOLE　,,,0)

コード	機　能	説　明
0	ノーマル	通常の表示
1	シークレット	文字は表示されない
2	ブリンク	点滅する
3	シークレット	「1」と同じ
4	リバース	反転する
5	リバースシークレット	反転して文字は表示されない
6	リバースブリンク	反転して点滅する
7	リバースシークレット	「5」と同じ

(カラーモードの場合／CONSOLE　,,,1)

コード	機能
0	黒
1	青
2	赤
3	紫
4	緑
5	水色
6	黄色
7	白

・バックグラウンドカラーとフォアグラウンドカラーはグラフィック画面のカラーを設定するものです。ここでの説明は省略します。

・ボーダーカラーは，PC-8801mkIIでは意味を持ちません。通常は省略して使います。

〈書き方例〉

① CONSOLE ,,,0　　画面に「ABC」の文字をリバースで表示します。

```
CONSOLE ,,,0
COLOR 4
PRINT "ABC"
```

② CONSOLE ,,,1　　画面に「ABC」の文字を緑色で表示します。

```
CONSOLE ,,,1
COLOR 4
PRINT "ABC"
```

③ CONSOLE ,,,1　　画面に「カラー」の文字を黄色で表示します。

```
CONSOLE ,,,1
X=2:Y=4
COLOR X+Y
PRINT "カラー"
```

(BEEP命令の説明)

〈一般形式〉

BEEP　スイッチ

・内蔵スピーカー（ブザー）を鳴らします。

・スイッチには「0」か「1」を指定します。

・スイッチに「1」を指定すると，ブザーは鳴り続けます。

・スイッチに「0」を指定すると，ブザーは鳴り止みます。

・スイッチの指定を省略すると，一定の時間だけブザーが鳴ります。

〈書き方例〉

① BEEP　　一定時間ブザーが鳴ります。

```
BEEP
```

② BEEP 1　　FOR～NEXTタイマーが100回ループする間ブザーが鳴ります。

```
BEEP 1
FOR I=1 TO 100:NEXT
BEEP 0
```

9.4 音楽の演奏

PC-8801mkIIは，内蔵スピーカーを利用して音楽を演奏することができます。

音楽を演奏するには「CMD SING」という拡張命令を使います。拡張命令にはCMD SING以外にも次のものがあります。

① CMD TURTLE――――――タートルグラフィックで図形を描きます。
② CMD CLS――――――――画面を高速にクリアします。
③ CMD TEXT ON／OFF―――テキスト画面の表示を制御します。
④ CMD CUT ――――――――拡張命令の使用を中止します。

なお，これらの四つの命令は，下巻で説明することにします。

9.4.1 拡張命令を使う前の準備

拡張命令は，PC-8801mkIIを起動しただけでは使うことができません。拡張命令を使うためには，システムディスクに用意されている「拡張命令パッケージ」というユーティリティプログラムを実行して，拡張命令の機能をメモリーに読み込みます。

拡張命令パッケージ・ユーティリティは，システムディスクの中の「@load.n88」というBASICプログラムと「@exst＊」という機械語ファイルの二つのファイルから成り立っています。「@load.n88」というプログラムを実行すれば，CMD SINGと他の四つの拡張命令が使えるようになります。

設問 81 拡張命令パッケージ・ユーティリティをロードして実行して下さい。

【 実行結果例 】

```
load "@load.n88"
Ok
run
The following statements are available.

  1. CMD TURTLE
  2. CMD SING
  3. CMD CLS
  4. CMD TEXT ON/OFF
  5. CMD CUT

Please enjoy new features.
Ok
```

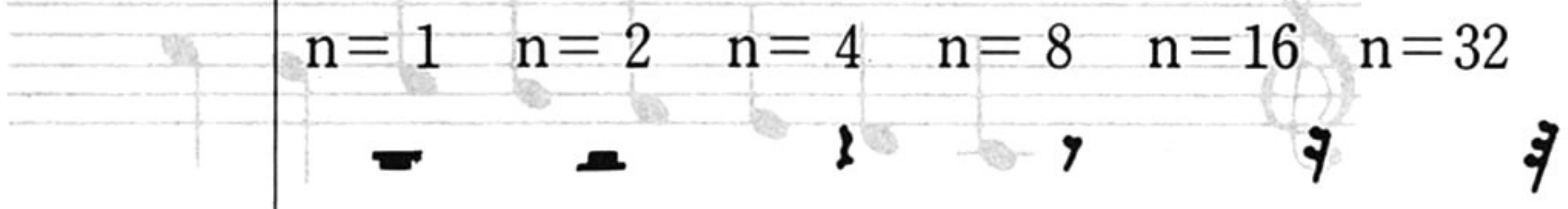
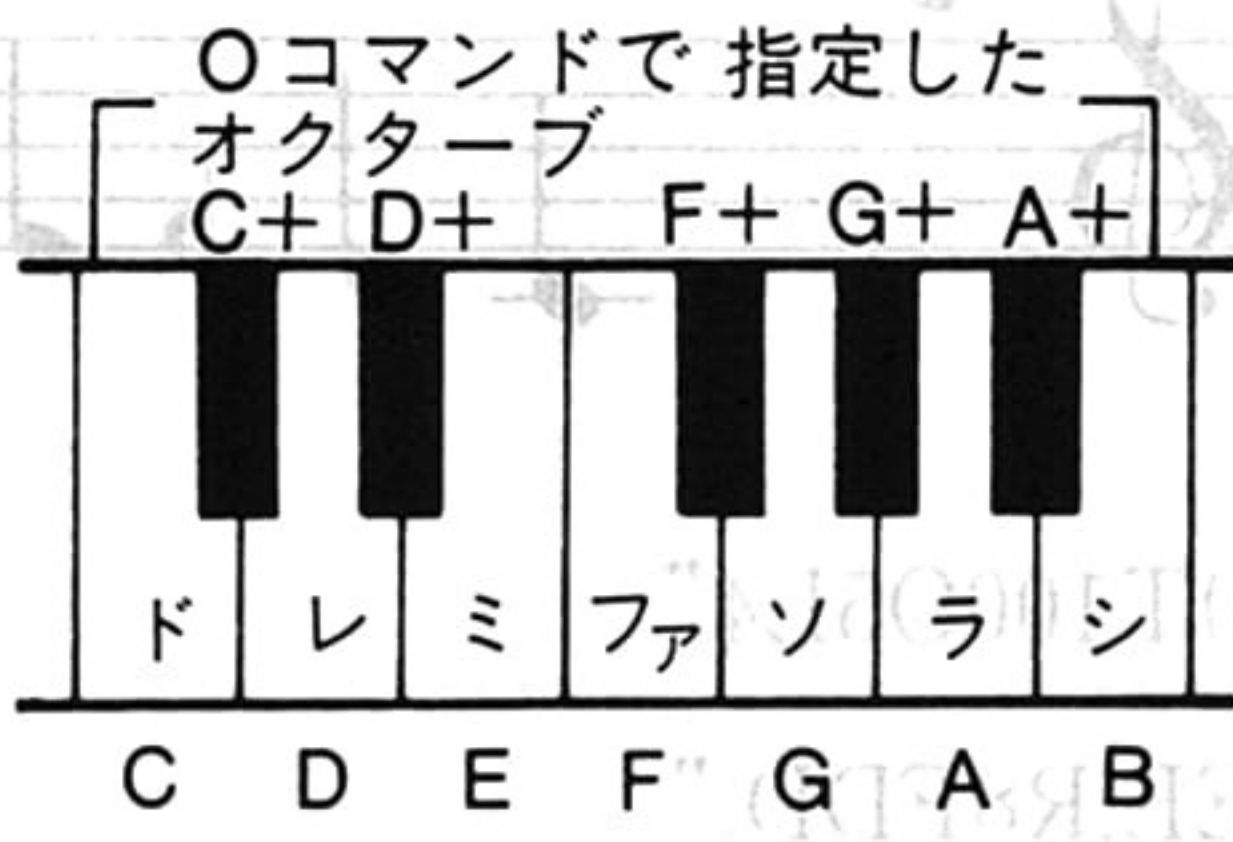
【 実行後の説明 】

・「@load.n88」を実行すると，機械語ファイル「@exst＊」を読み込み，メモリーに拡張命令を定義します。

・これで，五つの拡張命令が使えるようになりました。

9.4.2 音楽を奏でるには

音楽を演奏するCMD SING命令の説明を次に示します。

(CMD SING命令の説明)

〈一般形式〉

CMD SING　サブコマンドストリング

・サブコマンドストリングで指定したメロディーを演奏します。

・サブコマンドストリングには，ミュージックサブコマンドを並べた文字列を指定します。

・ミュージックサブコマンドには，次のものがあります。

コマンド	条　件	機　　能
Tn	48≦n≦225	演奏するテンポを設定します。nの値が大きいほど速くなります。nの値は1分間に演奏する4分音符の数を表します。
On	3≦n≦6	オクターブを設定します。nの値とオクターブとの対応は，次のとおりです。
Nn	1≦n≦32	音符の長さを設定します。nには，音符の種類を指定します。 n＝1　n＝2　n＝4　n＝8　n＝16　n＝32 全音符　2分音符　4分音符　8分音符　16分音符　32分音符
Rn.	1≦n≦32	休符を表します。nには，休符の種類（長さ）を指定します。符点休符には「.(ピリオド)」を付けます。nを省略した場合は，「Ln」と同じ値を指定したものとみなします。

コマンド	条　件	機　　能
		n＝1　n＝2　n＝4　n＝8　n＝16　n＝32 全休符 2分休符 4分休符 8分休符 16分休符 32分休符
C～B ±n.	1≦n≦32	音符を表します。C～Bで対応する音符を指定します。「＋」を指定すると半音上がり，「－」を指定すると半音下がります。 Oコマンドで 指定した オクターブ C+ D+ F+ G+ A+ ド レ ミ ファ ソ ラ シ C D E F G A B nには，音符の長さを指定します。符点音符には「.(ピリオド)」を付けます。
Xn	n＝0　または n＝1	テキスト画面の表示を制御します。「X0」で，テキスト画面の表示を一時的に消します。「X1」では，画面は消えません。澄んだ音を出すときは，「X0」を指定します。
RPn […]	1≦n≦255	「[　]」で囲まれたストリングをn回繰り返します。「[　]」の入れ子（ネスティング）は8回まで可能です。

(注)　サブコマンドは，小文字で指定してもかまいません。

・ミュージックサブコマンドに指定するパラメータ（「n」の部分）は，整数か変数で与えます。

・ミュージックサブコマンドのパラメータを変数で指定するときは，その変数を「( )」で囲みます。

・音の高さ，長さなどは，テキスト画面を消したとき（「X0」のとき）を基準に設定してありますが，多少の誤差があります。

〈書き方例〉

① CMD　SING　"X0T120O4L4CDEFGABO5C"

テキスト画面を消し，テンポ120，オクターブ4で4分音符で「ドレミファソラシド」を演奏します。

♩＝120

② CMD　SING　"X0T120O4L4CDE-FGA-B-O5C"

テキスト画面を消し，テンポ120，オクターブ4で，4分音符で「ドレミファソラシド」を短音階（「ミ」，「ラ」，「シ」を半音下げる）で演奏します。

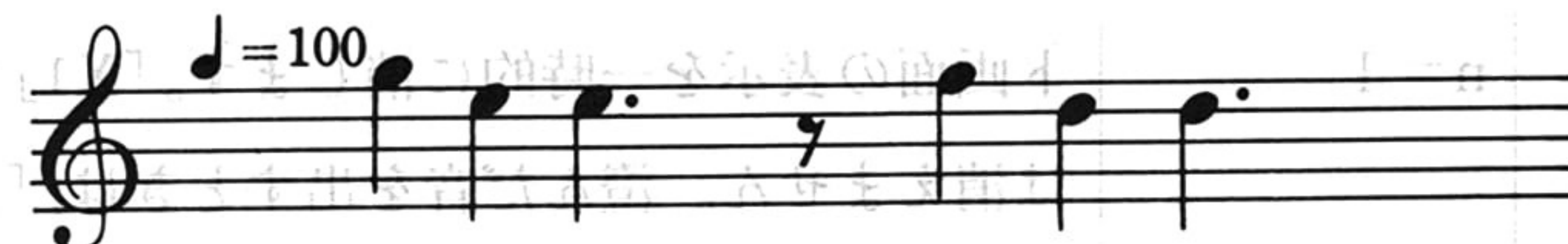
♩＝120

③ A$＝"X0T100O5L4"

B$＝"GEE.R8FDD."

CMD　SING　A$＋B$

テキスト画面を消し，テンポ100，オクターブ5で，4分音符を中心に「ソミミファレレ」を演奏します。

♩＝100

④ PT＝100：PO＝5

X$＝"X0T(PT)O(PO)L4"

Y$＝"RP2[CDE.R8]"

CMD　SING　X$＋Y$

テキスト画面を消し，テンポ100，オクターブ5で，「ドレミ」を2回繰り返します。

♩＝100

9.4.3 プログラム例(9)———音楽（「草競馬」）を演奏するプログラム

設問 82　メモリークリアした後，次のプログラムをキーインして実行して下さい。

【 プログラムリスト 】

```
10 ' ｸｻｹｲﾊﾞ
20 PTEM=135:POC1=4:POC2=5:PL=8
30 A$="AAAF+A16A16B16B16AF+R"
40 B$="AAAF+AB16B16A16A16F+R"
50 C$="AAAF+16F16A16A16BAF+R"
60 E$="D.D16F+AO(POC2)D4.O(POC1)R"
70 F$="B.B16O(POC2)DO(POC1)BA4."
80 G$="F+16G16AAF+16F+16A16A16BAF+4"
90 H$="EF+16A16F+16E.D4.R"
100 L$="F+E4.F+E4."
110 M$="E4F+ED4R"
120 N$="EF+ED4.R"
130 CMD SING "X0T(PTEM)O(POC1)L(PL)"
140 CMD SING A$+L$+B$+M$
150 CMD SING C$+L$+C$+N$
160 CMD SING E$+F$+G$+H$
170 END
```

【 実行後の説明 】

・このプログラムは「草競馬」という曲を演奏するプログラムです。

・このプログラムは，次に示す楽譜から作成したものです。

草競馬

♩=135

・このプログラムの実行中は，テキスト画面が一時的に消えます。

・音量（ボリューム）は，本体前面の扉を開けて「ブザー音量ボリューム（VOLUME）」をマイナスドライバーを使って調整します。

練習問題

【 基本的なプログラムの作成 】

問題 1　キーボードから得点を入力し，0～100点の間であれば入力した得点を，それ以外であれば「エラー」の文字を画面に表示するプログラムを作りなさい。

問題 2　下記の資料を基に，A製品の最高売上数と最低売上数を求め，最高と最低について，その月名と売上数を表示するプログラムを作成しなさい。

〈売上表〉

1月	2月	3月	4月	5月	6月	7月	8月	9月	10月	11月	12月
25	35	38	48	65	32	55	61	43	26	58	47

(処理条件)

①月と売上数は，一つのINPUT命令で入力します。

②月と売上数の入力命令には，次のプロンプト文を付けます。

ツキ，ウリアゲ

③画面クリア後，結果を表示します。

④結果の表示は次のようにします。

```
＊＊＊ケッカ　ヒョウジ＊＊＊
                          } 1行改行
＊サイコウ＊
ツキ　　　　XX
ウリアゲ　　XX
＊サイテイ＊
ツキ　　　　XX
ウリアゲ　　XX
```

ただし，Xは任意の文字

【 配列 】

問題 3　5人分の名前と成績を配列（1～5）に入力し，配列の内容とその平均点をプリンタに印字するプログラムを作りなさい。

問題 4　クラスの人数によって配列の個数を設定し，その人数分の成績を配列に入力して平均点を画面に表示するプログラムを作りなさい。

なお，この処理は終了の内容が入力されるまで繰り返すようにします。

問題 5　任意の正数を入力して，「1」から入力した数までの合計を画面に表示するプログラムを作りなさい。

【ソート】

問題 6　次の処理を行うプログラムを作成して下さい。

（処理）●名前，身長，体重を10人分入力し，身長の高い順にソートして，ソートした順に名前と身長および体重を表示します。

問題 7　問題6のプログラムを，体重の軽い順にソートするように変更して下さい。

問題 8　次のプログラムの (1) ～ (5) の中を埋めてプログラムを完成させなさい。

```
10 DIM (1)
20 FOR X=0 TO 9
30   A(X)=INT(RND(1)*100)
40 NEXT X
50 FOR Y=8 TO 0 (2)
60   FOR X=0 TO Y
70     IF A(X)>A(X+1) THEN (3)
80 (4)
90 FOR X=0 TO 9
100  PRINT A(X);
110 NEXT X: (5)
120 END
```

（隣接交換法　一方向型）

問題 9　次のプログラムについて以下の問に答えて下さい。

（プログラムリスト）

```
10 DIM CD(5),NM$(5)
20 DATA 29,ｺﾊﾞﾔｼ,2,ﾀｶﾊｼ,24,ｵｵﾉ,8,ﾔﾏﾓﾄ,6,ｺﾊﾞﾔｶﾜ
30 FOR I=1 TO 5
40   READ CD(I),NM$(I)
50 NEXT I
```

〈問1〉このプログラムのデータを，コードの小さい順にソートして，さらにプリンタに印字するように，命令を追加して下さい。

〈問2〉このプログラムのデータを，名前が五十音順になるようにソートして，さらにプリンタに印字するように，命令を追加して下さい。

問題 10　次の処理概要を読んで，プログラムの空欄を埋めて下さい。

（処理概要）●配列に出席番号と名前をキーボードから入力し，出席番号順にソートして出席番号と名前を画面に表示します。

●最大10人分のデータが処理できます。

●出席番号として「999」を入力するとデータ入力の処理からソートの処理へ移ります。

（プログラムリスト）

```
10 DIM NO(9), [ (1) ]
20 FOR I=0 TO 9
30   INPUT "ﾊﾞﾝｺﾞｳ =";X
40   IF X=999 THEN 70 ELSE [ (2) ]
50   INPUT "ﾅﾏｴ =";NM$(I)
60 NEXT I
70 I=I-1
80 FOR J=I-1 TO 0 STEP -1
90   FOR K=0 TO J
100    IF NO(K)>NO(K+1) THEN GOSUB [ (3) ]
110 [ (4) ]
120 FOR L=0 TO I
130   PRINT NO(L);NM$(L)
140 NEXT L
150 END
160 SWAP NO(K),NO(K+1)
170 [ (5) ]
180 RETURN
```

【 棒グラフ作成 】

問題 11　次の処理を行うプログラムを条件に従って作成して下さい。

（処　理）●ある営業所の半期の売上を月ごとに入力し，合計をとっておきます。

●合計に対する各月の売上の割合を棒グラフの形にして，画面に表示します。

（条　件）●入力は月と売上の２項目とします。

●入力したデータは，売上の合計をとりながら配列へ入れます。

●入力項目を配列へ入れる場合は，〔FOR～NEXT〕を使用します。

これは，FOR～NEXTのカウンタの値を配列の添字として利用するためです。

●合計に対する各月の売上の100分比を求め，その比を横方向の棒グラフで表示し

ます。

●棒グラフは，グラフィックキャラクタの"■"を使って〔FOR～NEXT〕で表示します。

【 サーチ 】

問題 12　次のプログラムの処理概要を読んで，プログラムリストの空欄を埋めて下さい。

(処理概要) ●DATA文で支店コードと支店名を定義しておき，配列へ読み込みます。

●キーボードから支店コードを入力して，対応する支店名を画面に表示します。

(プログラムリスト)

```
10 DIM CD(7),NM(7)
20 DATA 10,Tokyo,20,Sapporo,30,Nigata,40,Nagoya
30 DATA 50,Osaka,60,Kokura,70,Fukuoka,80,Kagoshima
40 [(1)          ]
50   READ CD(I),NM$(I)
60 NEXT I
70 INPUT "ｼﾃﾝ ｺｰﾄﾞ = ";[(2)    ]
80 FOR J=0 TO 7
90   IF IC = [(3)  ] THEN 110
100 [(4)    ]
110 PRINT TAB(13);[(5)    ];"ｼﾃﾝ"
120 END
```

問題 13　次のプログラムは，社員番号，名前，出身地の3項目を配列に読み込むものです。このプログラムに次の処理を追加して下さい。

(追加処理) キーボードから社員番号を入力して，対応する名前と出身地を画面に表示する。

(条件) ●社員番号を入力する時に，画面へ"シャインNo.="というメッセージを表示すること。

●社員番号として，999が入力されたとき処理を終了すること。

●入力された社員番号を配列内で探すときは，〔SEARCH〕関数を使うこと。

●配列内に存在しない社員番号が入力されたときは，"データガ　アリマセン"というメッセージを画面に表示すること。

●名前と出身地を表示するときは，その前に"(ナマエ)=","(シュッシン)="というメッセージをつけること。

（プログラムリスト）

```
10 DATA 68,"ｶｻｲ ﾐﾁﾋｺ","ﾅｺﾞﾔ"
20 DATA 47,"ｲﾉｳｴ ﾖｳｺ","ﾋｮｳｺﾞ"
30 DATA 64,"ﾊﾗ ﾀｶﾕｷ","ﾅｶﾞﾉ"
40 SU=3:DIM B%(SU),N$(SU),S$(SU)
50 FOR I=1 TO SU
60   READ B%(I),N$(I),S$(I)
70 NEXT I
```

【 多重ループ 】

問題 14　1から100までの整数を，次のようにプリンタに印字して下さい。

```
  1   2   3   4   5   6   7   8   9  10
 11  12  13  14  15  16  17  18  19  20
 21  22  23  24  25  26  27  28  29  30
 31  32  33  34  35  36  37  38  39  40
 41  42  43  44  45  46  47  48  49  50
 51  52  53  54  55  56  57  58  59  60
 61  62  63  64  65  66  67  68  69  70
 71  72  73  74  75  76  77  78  79  80
 81  82  83  84  85  86  87  88  89  90
 91  92  93  94  95  96  97  98  99 100
```

【 シーケンシャルファイル 】

問題 15　次に示す処理を行うプログラムを作成して下さい。

（処理）●商品コード，商品名，単価の3項目をキーボードから入力して，ドライブ2の「shohin」というファイルに出力する。

（条件）●商品コードと単価は数値で入力するものとします。

●商品コードとして999が入力されたとき，処理を終了させます。

（ヒント）●ファイルのOPENは，新しくファイルを作るときのモード（OUTPUT）で行います。

●処理を終了させるときは，ファイルをCLOSEします。

問題 16 問題15のプログラムで「shohin」というファイルに出力したデータを画面に表示するプログラムを，次の条件で作成して下さい。

（条件）●表示する項目は，商品コード，商品名，単価の３項目です。

●各項目を表示する位置は，TAB関数を使って合わせるようにします。

●各項目の見出しを表示します。

●画面に表示するレイアウトは次のとおりとします。

```
 2            13                 27
 ショウヒンコード  ショウヒンメイ        タ ン カ
 ×――――――×  ×――――――――――×  ¥99,999
      ↓             ↓              ↓     （編集）
 ×――――――×  ×――――――――――×  ¥99,999
```

（単価の表示は，「¥」記号と「,」記号で編集すること）

（ヒント）●シーケンシャルファイルからデータを読むときのOPEN命令は，INPUTモードで行います。

●読むべきデータがなくなったかどうかは，EOF関数で判定します。

●シーケンシャルファイルのデータをプログラムの変数に読み込むためには，「INPUT#」命令を使います。

●ファイルのクローズは，「CLOSE」命令で行います。

問題 17 次のプログラムは，試験の得点をドライブ２の「data」というシーケンシャルファイルに出力するものです。このプログラムに関する，以下の問に答えて下さい。

(プログラムリスト)

```
10 [(1)] "2:data" FOR [(2)] AS [(3)]
20 CLS:PRINT "*******  Input Data  *******"
30 FOR I=1 TO 5
40   INPUT "---- ﾅﾏｴ ----"; [(4)]
50   INPUT "ﾄｸﾃﾝ  ｺｸｺﾞ  =";KO
60   INPUT "     ｽｳｶﾞｸ =";SU
70   INPUT "     ｴｲｺﾞ  =";EI
80   [(5)] #1,NM$,KO,SU,EI
90   PRINT
100 NEXT I
110 CLOSE #1:END
```

〈問1〉 プログラムの中の空欄を埋めて下さい。

〈問2〉 このプログラムは，5人分の処理を行うと終了します。そこで，何人もの処理ができるように，名前に "end" と入力されたら処理を終了するようにして下さい。

〈問3〉 このプログラムは，新しくファイルを作るモード (OUTPUT) で，ファイルをOPENしています。これを，データの追加を行うモード (APPEND) で行うようにして下さい。

〈問4〉 このプログラムで作成したデータをメモリーに読み込んでプリンタに印字するプログラムを作成して下さい。

【 サブルーチン 】

問題 18 次のプログラムの (1) ～ (5) の中を埋めてプログラムを完成させなさい。

(プログラムリスト)

```
10 DIM A(10)
20 MAX=0:MIN=999
30 FOR N=1 TO 10
40   INPUT "ﾃﾝｽｳ (0-100) =";A(N)
50 [(1)]
60 FOR N=1 TO 10
70   GOSUB *MAXIMUM
80   [(2)]
90 NEXT N
100 PRINT "ｻｲﾀﾞｲ";TAB(10);MAX
110 PRINT "ｻｲｼｮｳ";TAB(10);MIN
120 [(3)]
130 *MAXIMUM
```

(つづく)

```
140 IF A(N)>MAX THEN MAX=A(N)
150 [ (4) ]
160 *MINIMUM
170 IF A(N)< [ (5) ] THEN MIN=A(N)
180 RETURN
```

【サーチとシーケンシャルファイル】

問題 19 次の処理を行うプログラムを作成して下さい。

(処理) ●商品の売上をシーケンシャルファイルを使って管理します。

(条件) ●商品コード，商品名，単価，数量の4項目で売上を管理します。

●商品コード，商品名，単価はあらかじめ配列に登録しておきます。

●キーボードから入力する項目は，商品コードと数量です。

●商品コードとして「999」が入力されたら，いままで入力したデータをドライブ2の「uriage」というファイルへ，商品コード，商品名，単価，数量の4項目を1件の形（レコード）として全部出力し，処理を終了します。

●商品コードが入力されたら，該当する商品名と単価を表示したのち，数量を入力します。

●数量が入力されたら，単価と掛け合わせて金額を算出し，表示します。

●以下のように商品コードと商品名および単価を決めます。

配列の添字	1	2	3	4	5	6
商品コード	14	23	29	10	18	50
商　品　名	ペンダント	ユビワ	イヤリング	ネックレス	ブレスレット	ソ　ノ　タ
単　　価	1,800	2,500	1,500	3,000	2,000	—

●商品コードとして「50(ソノタ)」が入力されたら，単価も入力するようにします。

【関　数】

問題 20 次に示す処理を行うプログラムを作成して下さい。

(処理) ●2桁以内（0～99の範囲）の乱数を発生させて，順に表示します。

(条件) ●発生させる乱数は5個です。

●表示は横に5個並べます。

(ヒント) ●乱数を発生させるにはRND関数を使います。

●乱数は0.00001～0.99999の範囲で発生するので，求める値は発生した乱数を100倍した後，整数型に変えます。

●実数型の値を整数型の値に変換するにはFIX関数かINT関数を使います。

●乱数は配列に入れた後，表示します。

●横に並べて表示するにはPRINT命令で「；（セミコロン）」を使います。

問題 21　自分の名前を入力し，"ワタシハ　サトウ　デス"の文字列から，ワタシハとデスを取り出し，入力した名前と合わせて3行になるように画面に表示するプログラムを作りなさい。

（表示例）

```
ワタシハ
ナカハタ
デ゛ス
```

【 漢字 】

問題 22　次の条件で漢字を表示して下さい。

（条件）●表示する漢字　"教育講座シリーズ"

●表示する位置　(150, 120)から右方向へ表示

●文字の色　青

●背景色　黄色

●グラフィック画面はカラーモード

問題 23　次の文字をプリンタに印字するプログラムを作って下さい。

（印字する文字）

"漢字の練習"

問題 24　問題22で表示した漢字を，プリンタに印字するプログラムを作って下さい。

付　　録

付.1　練習問題の解答

問題１の解答例

```
10 INPUT A
20 IF A<0 THEN 50
30 IF A>100 THEN 50
40 PRINT A:END
50 PRINT "ｴﾗｰ":END
```

問題２の解答例

```
10 HU=0:LU=100
20 IF CNT=12 THEN 70
30 INPUT "ﾂｷ,ｳﾘｱｹﾞ ";NT,NU
40 IF NU>HU THEN HT=NT:HU=NU
50 IF NU<LU THEN LT=NT:LU=NU
60 CNT=CNT+1:GOTO 20
70 CLS
80 PRINT "***ｹｯｶ ﾋｮｳｼﾞ***"
90 PRINT
100 PRINT "*ｻｲｺｳ*"
110 PRINT "ﾂｷ     ";HT
120 PRINT "ｳﾘｱｹﾞ ";HU
130 PRINT "*ｻｲﾃｲ*"
140 PRINT "ﾂｷ     ";LT
150 PRINT "ｳﾘｱｹﾞ ";LU
160 END
```

問題３の解答例

```
10 DIM NAMAE$(5),SEISEKI(5)
20 X=1:Y=1
30 IF X>5 THEN 80
40 INPUT "ﾅﾏｴ  =";NAMAE$(X)
50 INPUT "ｾｲｾｷ =";SEISEKI(X)
60 PRINT
70 X=X+1:GOTO 30
80 IF Y>5 THEN 120
90 LPRINT NAMAE$(Y);TAB(10);SEISEKI(Y)
100 HEIKIN=HEIKIN+SEISEKI(Y)
110 Y=Y+1:GOTO 80
120 HEIKIN=HEIKIN/5
130 LPRINT
140 LPRINT "ﾍｲｷﾝ";TAB(10);HEIKIN
150 END
```

問題 4 の解答例

入力した人数は，そのまま配列の宣言の個数に使います。

処理を数回繰り返すために，「ERASE」を使います。

```
10 INPUT "ﾆﾝｽﾞｳ =";NIN
20 DIM SEISEKI(NIN)
30 X=1:Y=1
40 IF X>NIN THEN 80
50 INPUT "ｾｲｾｷ =";SEISEKI(X)
60 PRINT
70 X=X+1:GOTO 40
80 HEIKIN=0
90 IF Y>NIN THEN 120
100 HEIKIN=HEIKIN+SEISEKI(Y)
110 Y=Y+1:GOTO 90
120 PRINT "ﾍｲｷﾝ";TAB(10);HEIKIN/NIN
130 PRINT
140 INPUT "ﾂﾂﾞｸ(1)?  ｵﾜﾘ(2)";A
150 IF A=1 THEN PRINT:ERASE SEISEKI:GOTO 10
160 IF A=2 THEN END
170 PRINT "ｴﾗｰ":GOTO 140
```

問題 5 の解答例

```
10 INPUT "ｶｽﾞ =";N
20 FOR X=1 TO N
30   GOKEI=GOKEI+X
40 NEXT X
50 PRINT "ｺﾞｳｹｲ";TAB(8);GOKEI
60 END
```

問題 6 の解答例

```
10 DIM NM$(10),SI(10),TA(10)
20 FOR I=1 TO 10
30   INPUT "ﾅﾏｴ =";NM$(I)
40   INPUT "ｼﾝﾁｮｳ =";SI(I)
50   INPUT "ﾀｲｼﾞｭｳ =";TA(I):PRINT
60 NEXT I
70 FOR I=1 TO 9
80   FOR J=I+1 TO 10
90     IF SI(I)<SI(J) THEN GOSUB 200
100 NEXT J,I
110 PRINT "ﾅﾏｴ";TAB(15);"ｼﾝﾁｮｳ";TAB(25);"ﾀｲｼﾞｭｳ"
```

(つづく)

```
120 FOR I=1 TO 10
130   PRINT NM$(I);TAB(14);SI(I);TAB(24);TA(I)
140 NEXT I
150 END
200 SWAP NM$(I),NM$(J)
210 SWAP SI(I),SI(J)
220 SWAP TA(I),TA(J)
230 RETURN
```

問題 7 の解答例

(修正例)

```
90     IF TA(I)>TA(J) THEN GOSUB 200
```

(プログラムリスト)

```
10 DIM NM$(10),SI(10),TA(10)
20 FOR I=1 TO 10
30   INPUT "ﾅﾏｴ =";NM$(I)
40   INPUT "ｼﾝﾁｮｳ =";SI(I)
50   INPUT "ﾀｲｼﾞｭｳ =";TA(I):PRINT
60 NEXT I
70 FOR I=1 TO 9
80   FOR J=I+1 TO 10
90     IF TA(I)>TA(J) THEN GOSUB 200
100 NEXT J,I
110 PRINT "ﾅﾏｴ";TAB(15);"ｼﾝﾁｮｳ";TAB(25);"ﾀｲｼﾞｭｳ"
120 FOR I=1 TO 10
130   PRINT NM$(I);TAB(14);SI(I);TAB(24);TA(I)
140 NEXT I
150 END
200 SWAP NM$(I),NM$(J)
210 SWAP SI(I),SI(J)
220 SWAP TA(I),TA(J)
230 RETURN
```

問題 8 の解答例

2 桁以内の乱数を発生させて，昇順（小→大）に並べ替え，画面に横に表示するプログラムです。

```
(1)   A(9)
(2)   STEP -1
(3)   SWAP A(X),A(X+1)
(4)   NEXT X,Y
(5)   PRINT
```

問題９の解答例

・〈問１〉の修正例

```
60 FOR I=1 TO 4
70   FOR J=I+1 TO 5
80     IF CD(I)>CD(J) THEN GOSUB 150
90 NEXT J,I
100 FOR I=1 TO 5
110   PRINT CD(I);NM$(I)
120 NEXT I
130 END
150 SWAP CD(I),CD(J)
160 SWAP NM$(I),NM$(J)
170 RETURN
```

（プログラムリスト）

```
10 DIM CD(5),NM$(5)
20 DATA 29,ｺﾊﾞﾔｼ,2,ﾀｶﾊｼ,24,ｵｵﾉ,8,ﾔﾏﾓﾄ,6,ｺﾊﾞﾔｶﾜ
30 FOR I=1 TO 5
40   READ CD(I),NM$(I)
50 NEXT I
60 FOR I=1 TO 4
70   FOR J=I+1 TO 5
80     IF CD(I)>CD(J) THEN GOSUB 150
90 NEXT J,I
100 FOR I=1 TO 5
110   PRINT CD(I);NM$(I)
120 NEXT I
130 END
150 SWAP CD(I),CD(J)
160 SWAP NM$(I),NM$(J)
170 RETURN
```

・〈問２〉の修正例

```
60 FOR I=1 TO 4
70   FOR J=I+1 TO 5
80     IF NM$(I)>NM$(J) THEN GOSUB 150
90 NEXT J,I
100 FOR I=1 TO 5
110   PRINT CD(I);NM$(I)
120 NEXT I
130 END
150 SWAP CD(I),CD(J)
160 SWAP NM$(I),NM$(J)
170 RETURN
```

（プログラムリスト）

```
10 DIM CD(5),NM$(5)
20 DATA 29,ｺﾊﾞﾔｼ,2,ﾀｶﾊｼ,24,ｵｵﾉ,8,ﾔﾏﾓﾄ,6,ｺﾊﾞﾔｶﾜ
30 FOR I=1 TO 5
40   READ CD(I),NM$(I)
50 NEXT I
60 FOR I=1 TO 4
70   FOR J=I+1 TO 5
80     IF NM$(I)>NM$(J) THEN GOSUB 150
90 NEXT J,I
100 FOR I=1 TO 5
110   PRINT CD(I);NM$(I)
120 NEXT I
130 END
150 SWAP CD(I),CD(J)
160 SWAP NM$(I),NM$(J)
170 RETURN
```

（解説）

文字型同士を比較すると，先頭から１文字ずつ文字コード（文字コード表参照）で比較していきますので，ソート後のデータは文字コードの順に並べ替えられています。

問題10の解答例

```
(1)   NM$(9)
(2)   NO(I)=X
(3)   160
(4)   NEXT K,J
(5)   SWAP NM$(K),NM$(K+1)
```

問題11の解答例

```
10 DIM TSUKI%(5),KINGAKU(5)
20 FOR I=0 TO 5
30   INPUT "ﾅﾝｶﾞﾂ ﾉ ｳﾘｱｹﾞﾃﾞｽｶ ";TSUKI%(I)
40   INPUT "  ｳﾘｱｹﾞ ｶﾞｸ ﾊ      ";KINGAKU(I)
50   GOKEI#=GOKEI#+KINGAKU(I)
60 NEXT I
70 PRINT:PRINT
80 FOR I=0 TO 5
90   PER=INT(KINGAKU(I)*100/GOKEI#)
100  PRINT USING " ## ｶﾞﾂ ";TSUKI%(I);
110  FOR J=1 TO PER
120    PRINT "■";
130  NEXT J
140  PRINT USING "  ### %";PER
150 NEXT I
160 END
```

問題12の解答例

```
(1)   FOR I=0 TO 7
(2)   IC
(3)   CD(J)
(4)   NEXT J
(5)   NM$(J)
```

問題13の解答例

```
80 INPUT "ｼｬｲﾝ NO. = ";BAN
90 IF BAN=999 THEN END
100 J=SEARCH(B%,BAN)
110 IF J=-1 THEN PRINT "ﾃﾞｰﾀ ｶﾞ ｱﾘﾏｾﾝ !!":GOTO 80
120 PRINT "(ﾅﾏｴ) = ";N$(J);
130 PRINT TAB(20);"(ｼｭｯｼﾝ) = ";S$(J)
140 PRINT:GOTO 80
```

(プログラムリスト)

```
10 DATA 68,"ｶｻｲ ﾐﾁﾋｺ","ﾅｺﾞﾔ"
20 DATA 47,"ｲﾉｳｴ ﾖｳｺ","ﾋｮｳｺﾞ"
30 DATA 64,"ﾊﾗ ﾀｶﾕｷ","ﾅｶﾞﾉ"
40 SU=3:DIM B%(SU),N$(SU),S$(SU)
50 FOR I=1 TO SU
60   READ B%(I),N$(I),S$(I)
70 NEXT I
80 INPUT "ｼｬｲﾝ NO. = ";BAN
90 IF BAN=999 THEN END
100 J=SEARCH(B%,BAN)
110 IF J=-1 THEN PRINT "ﾃﾞｰﾀ ｶﾞ ｱﾘﾏｾﾝ !!":GOTO 80
120 PRINT "(ﾅﾏｴ) = ";N$(J);
130 PRINT TAB(20);"(ｼｭｯｼﾝ) = ";S$(J)
140 PRINT:GOTO 80
```

問題14の解答例

```
10 FOR J=0 TO 90 STEP 10
20   FOR I=1 TO 10
30     LPRINT USING "###  ";I+J;
40   NEXT I:LPRINT:LPRINT
50 NEXT J
60 END
```

１行中に10個の数値を続けて印字するため，行番号40は「；（セミコロン)」で終わります。行番号60の二つの「LPRINT」命令は，改行をします。

問題15の解答例

```
10 OPEN "2:shohin" FOR OUTPUT AS #1
20 INPUT "ｼｮｳﾋﾝ ｺｰﾄﾞ =";CD
30 IF CD=999 THEN 80
40 INPUT "ｼｮｳﾋﾝ ﾒ ｲ  =";NM$
50 INPUT "  ﾀ  ﾝ  ｶ  =";TK
60 WRITE #1,CD,NM$,TK
70 GOTO 20
80 CLOSE #1:END
```

問題16の解答例

```
10 OPEN "2:shohin" FOR INPUT AS #1
20 PRINT TAB(1);"ｼｮｳﾋﾝｺｰﾄﾞ";TAB(12);
30 PRINT "ｼｮｳﾋﾝ ﾒ ｲ";TAB(29);"ﾀ ﾝ ｶ"
40 IF EOF(1) THEN 90
50 INPUT #1,CD,NM$,TK
60 PRINT CD;TAB(12);NM$;
70 PRINT USING "¥¥##,###";TK
80 GOTO 40
90 CLOSE:END
```

問題17の解答例

・〈問１〉の解答

```
(1)   OPEN
(2)   OUTPUT
(3)   #1
(4)   NM$
(5)   WRITE
```

・〈問２〉の修正例

```
30 ──────→削除
45 IF NM$="end" THEN 110
100 GOTO 40
```

(プログラムリスト)

```
10 OPEN "2:data" FOR OUTPUT AS #1
20 CLS:PRINT "*******  Input Data  *******"
40   INPUT "---- ﾅﾏｴ ----";NM$
45 IF NM$="end" THEN 110
50   INPUT "ﾄｸﾃﾝ  ｺｸｺﾞ  =";KO
60   INPUT "      ｽｳｶﾞｸ =";SU
70   INPUT "      ｴｲｺﾞ  =";EI
80   WRITE #1,NM$,KO,SU,EI
90   PRINT
100 GOTO 40
110 CLOSE #1:END
```

・〈問３〉の修正例

```
10 OPEN "2:data" FOR APPEND AS #1
```

(プログラムリスト)

```
10 OPEN "2:data" FOR APPEND AS #1
20 CLS:PRINT "*******  Input Data  *******"
40   INPUT "---- ﾅﾏｴ ----";NM$
45 IF NM$="end" THEN 110
50   INPUT "ﾄｸﾃﾝ  ｺｸｺﾞ  =";KO
60   INPUT "      ｽｳｶﾞｸ =";SU
70   INPUT "      ｴｲｺﾞ  =";EI
80   WRITE #1,NM$,KO,SU,EI
90   PRINT
100 GOTO 40
110 CLOSE #1:END
```

・〈問４〉の解答例

```
10 OPEN "2:data" FOR INPUT AS #1
20 LPRINT "ﾅﾏｴ";TAB(16);"ｺｸｺﾞ";
30 LPRINT TAB(24);"ｽｳｶﾞｸ";TAB(31);"ｴｲｺﾞ"
40 IF EOF(1) THEN 80
50 INPUT #1,NM$,KO,SU,EI
60 LPRINT NM$;TAB(15);KO;TAB(23);SU;TAB(31);EI
70 GOTO 30
80 CLOSE #1:END
```

(解説)

シーケンシャルファイルの現在入っているデータに，新しいデータを追加したい場合はAPPENDモードでファイルをOPENすることにより行います。

次にシーケンシャルファイルをOPENするときの三つのモードについて簡単に説明します。

INPUTモード ……ファイルからデータを入力するときに使います。このモードで，データを出力することはできません。

OUTPUTモード……ファイルへデータを出力するときに使います。このモードでオープンすると，今までファイルに記録されていたデータは消えて，新たにデータを登録し直します。なお，このモードでデータを入力することはできません。

APPENDモード……ファイルデータを出力するときに使います。このモードはOUTPUTモードと違って，今までファイルに記録されていたデータをそのままにして，さらにその後に追加を行います。なお，このモードでデータを入力することはできません。

問題18の解答例

キーボードから10人分の点数を入力し，配列に記憶した後，最大の点数と最小の点数を見つけ出すプログラムです。

```
(1)    NEXT N
(2)    GOSUB *MINIMUM    または   GOSUB 160
(3)    END
(4)    RETURN
(5)    MIN
```

問題19の解答例

```
10 DATA 14,ペンダント,1800,23,ユビワ,2500
20 DATA 29,イヤリング,1500,10,ネックレス,3000
30 DATA 18,ブレスレット,2000,50,ソノタ,9999
40 X=6:DIM CD%(X),NM$(X),TA(X),SU(X)
50 FOR I=1 TO X
60   READ CD%(I),NM$(I),TA(I)
70 NEXT I
80 OPEN "2:uriage" FOR OUTPUT AS #1
90 INPUT "ショウヒン コード = ",SC%
100 IF SC%=999 THEN 190
110 J=SEARCH(CD%,SC%)
120 IF J=-1 THEN PRINT "ニュウリョク ミス!!":PRINT:GOTO 90
130 PRINT "       ナマエ  = ";NM$(J)
```

(つづく)

```
140 PRINT "      ﾀﾝｶ   = ";
150 IF SC%=50 THEN INPUT TA(J) ELSE PRINT TA(J)
160 INPUT "      ｽｳﾘｮｳ = ",SU(J):PRINT
170 PRINT "      ｷﾝｶﾞｸ = ";TA(J)*SU(J)
180 GOTO 90
190 FOR I=1 TO X
200   WRITE #1,CD(I),NM$(I),TA(I),SU(I)
210 NEXT I
220 CLOSE #1:END
```

問題20の解答例

```
10 DIM RS(4)
20 FOR I=0 TO 4
30   RS(I)=INT(RND(1)*100)
40 NEXT I
50 FOR J=0 TO 4
60   PRINT RS(J);
70 NEXT J:PRINT
80 END
```

問題21の解答例

```
10 A$="ﾜﾀｼﾊ ｻﾄｳ ﾃﾞｽ"
20 INPUT "ﾅﾏｴ =";NAMAE$
30 X$=LEFT$(A$,4)
40 Y$=RIGHT$(A$,3)
50 PRINT X$
60 PRINT NAMAE$
70 PRINT Y$
80 END
```

問題22の解答例

```
10 SCREEN 0:CLS 3
20 FOR X=150 TO 262 STEP 16
30   READ A$:B=VAL("&H"+A$)
40   PUT (X,120),KANJI(B),PSET,1,6
50 NEXT X
60 DATA 3635,3069,3956,3A42
70 DATA 2537,256A,213D,253A
```

（解説）

通常「PUT」命令で漢字を表示させるとき，表示しようとする位置に既に漢字が表示されている場合，重なったり指定したバックグラウンドカラーでぬりつぶされたりします。しかし，「PSET」または「PRESET」を指定すると，表示する場所がいったんクリアされてから表示されます。

問題23の解答例

```
10 LPRINT CHR$(&H1B);"K";
20 FOR I=1 TO 5
30   READ A$,B$
40   X=VAL("&H"+A$):Y=VAL("&H"+B$)
50   LPRINT CHR$(X);CHR$(Y);
60 NEXT I:LPRINT
70 LPRINT CHR$(&H1B);"H";
80 DATA 34,41
90 DATA 3B,7A
100 DATA 24,4E
110 DATA 4E,7D
120 DATA 3D,2C
```

問題24の解答例

```
10 LPRINT CHR$(&H1B);"K";
20 FOR I=1 TO 8
30   READ A$,B$
40   C=VAL("&H"+A$):D=VAL("&H"+B$)
50   LPRINT CHR$(C);CHR$(D);
60 NEXT I:LPRINT
70 LPRINT CHR$(&H1B);"H";
80 DATA 36,35,30,69,39,56,3A,42
90 DATA 25,37,25,6A,21,3D,25,3A
```

（解説）

プリンタへ漢字を印字する場合，このように「LPRINT CHR$ (&H 1 B)；"K"；」という命令を実行してから，漢字コードを印字します。これは，プリンタにこれ以降の「LPRINT」命令はすべて漢字を印字するものだ，ということを認識させるためです。また，漢字を印字し終わったら「LPRINT CHR$ (&H 1 B)；"H"；」という命令を実行して，漢字の印字が終了したことをプリンタに知らせなければなりません。

付.2　PC-8801mkⅡの主な関数

本文中で説明した関数のほかにもいろいろな関数があります。ここでは，それらの関数の中から業務処理等でよく使われる関数を説明します。

【 説明する関数 】

〈数値関数〉	〈文字関数〉	〈クロック関数〉
・FIX	・STR$	・TIME$
・SQR	・MIDS$	・DATE$
・SIN	・LEN	
・COS	・INSTR	
・LOG	・ASC	
・EXP	・HEX$	

このうちの「クロック関数」とは，時刻や日付を取り出したりセットしたりするものです。

(1)　FIX関数

FIX関数は，数値の小数部の切り捨てを行います。

(FIX関数の説明)

〈一般形式〉

FIX（数式）

・数式の値の小数点以下を切り捨てた値を求めます。

〈書き方例〉

①	PRINT　FIX(5.5)	画面に「5.5」の小数部を切り捨てた値「5」を表示します。
②	A=FIX(−8.3)	変数Aに「−8.3」を切り捨てた値「−8」を代入します。
③	N=3.14159 PRINT　FIX(N)	変数Nに代入されている「3.14159」の小数部を切り捨てた値「3」を画面に表示します。

(2)　SQR関数

SQR関数は，数値の平方根を求めます。

(SQR関数の説明)

〈一般形式〉

SQR（数式）

・数式の値の平方根を求めます。

〈書き方例〉

① A=SQR(２)　　変数Aに「２」の平方根の「1.41421」を代入します。

② B=5
PRINT　SQR(B)　　変数Bの内容の「５」の平方根「2.23607」を求めて，これを画面に表示します。

(3) SIN関数

SIN関数は，数値の正弦値(サイン)を求める三角関数です。

(SIN関数の説明)

〈一般形式〉

SIN（数式）

・数式の値は正弦値を求めます。
・数式の値は，ラジアンで指定します。
・角度をラジアンに変換するには，次の式を使います。

ラジアン値＝角度＊π／180

＝角度＊3.14159／180

〈書き方例〉

① A=SIN(30＊3.14159／180)　　sin30°の正弦値を求めて変数Aに代入します。

② N=3.14159／180
PRINT　SIN(90＊N)　　sin90°の正弦値を求めの画面に表示します。

(4) COS関数

COS関数は，数値の余弦値（コサイン）を求める三角関数です。

(COS関数の説明)

〈一般形式〉

COS（数式）

・数式の値の余弦値を与えます。
・数式の値はラジアンで指定します。

〈書き方例〉

① A=COS（30＊3.14159／180)　　COS30°の余弦値を求めて変数Aに代入します。

②	N＝3.14159／180 PRINT　COS(90＊N)	COS90°の余弦値を求めて画面に表示します。
③	N＝3.14159／180 X＝COS(30＊N)＊10 Y＝SIN(30＊N)＊10	半径10で角度30°のときの円周上の点の座標を求めます。

(5)　LOG関数

LOG関数は，自然対数を求めます。

(LOG関数の説明)

〈一般形式〉

LOG（数式）

・数式の値の自然対数（e＝2.718を底とした対数）を求めます。

・この関数で求められる値は，「$y=e^x$」の「x」に当たります。

〈書き方例〉

①	A＝LOG(10)	$\log_e 10$の自然対数を求め変数Aに代入します。
②	Y＝30 PRINT　LOG(Y)	$\log_e 30$の自然対数を求め画面に表示します。

(6)　EXP関数

EXP関数は，eに対する指数関数の値を求めます。

(EXP関数の説明)

〈一般形式〉

EXP（数式）

・e（＝2.718）に対する指数関数e^xの値を求めます。

・数式には，「e^x」の「x」に当たる値を指定します。

・数式に「87.33655」より大きい値を指定すると，「OV Error (Overflow)」が発生します。

〈書き方例〉

①	A＝EXP(2)	e^2の値「7.38906」を変数Aに代入します。
②	X＝5 PRINT　EXP(X)	e^5の値「148.413」を画面に表示します。

(7) STR$関数

STR$関数は，VAL関数と逆の機能を持っていて，数値を文字型データに変換します。

(STR$関数の説明)

〈一般形式〉

STR$（数式）

・数式の値を文字型データに変換します。

・数式が正の値のときは，変換されたデータは先頭に一文字分のスペースを含みます。

・数式が負の値のときは，変換されたデータは先頭に「－」を含みます。

〈書き方例〉

①	PRINT STR$(15)	「15」を文字変換して文字型3桁分の「 15」にし，画面に表示します。
②	A$＝STR$(－49)	「－49」を文字変換して文字型3桁の「－49」にし，変数A$に代入します。
③	N＝99 B$＝STR$(N)	「99」を文字変換して文字型3桁の「 99」にし，変数B$に代入します。

(8) MID$関数

MID$関数は，文字列の中から指定した長さの文字列を取り出す関数です。

(MID$関数の説明)

〈一般形式〉

MID$（文字列，数式1，数式2）

・文字列の中から数式1で示す位置から数式2で指定した長さの文字列を取り出します。

・数式1には，取り出す文字の初めの位置を指定します。1～255の範囲の値を指定することができます。

・数式2には，取り出す文字の桁数を指定します。0～255の範囲の値を指定することができます。

・数式2に，数式1で指定した位置から右側の文字数より大きな値が指定されたときは，数式1の位置から右側の文字列をすべて取り出します。

・数式2を省略すると，数式1の位置から右側の文字列をすべて取り出します。

・文字列の文字数が数式1の値より小さければ，ヌルストリングを与えます。

・数式1および数式2を小数で指定すると，小数点以下を四捨五入した整数として扱います。

〈書き方例〉

① PRINT MID$ ("ABCDEFG", 4, 3)

「ABCDEFG」という文字列の左から四番目の文字から3文字分の「DEF」を取り出し，画面に表示します。

② A$＝MID$ ("トウキョウ", 3, 2)

「トウキョウ」という文字列の左から3番目の文字から2文字分の「キョ」を取り出し，変数A$に代入します。

③ M$＝"オオサカ"：X＝3：Y＝1

B$＝MID$(M$, X, Y)

「オオサカ」という文字列の左から3番目の文字から1字分の「サ」を取り出し，変数B$に代入します。

④ N$＝"フクオカ"：Y＝2：Z＝8

C$＝MID$ (N$, Y, Z)

「フクオカ」という文字列の左から2番目の文字から8文字分，つまりここでは残り全部の文字列「クオカ」を取り出し，変数C$に代入します。

⑤ D$＝MID$ ("ヨコハマ", 3)

「ヨコハマ」という文字列の左から3番目の文字から右側の文字列すべて，つまりここでは文字列「ハマ」を取り出し，変数D$に代入します。

⑥ E$＝MID$ ("ナゴヤ", 9)

変数E$にヌルストリングを代入します。

(9) LEN関数

LEN関数は，文字列の長さを求めます。

(LEN関数の説明)

〈一般形式〉

LEN (文字列)

・文字列の総文字数を求めます。

・この関数は，画面やプリンタには表示されないコントロールコード(文字コード「1」～「31」の文字) なども文字数として数えます。

〈書き方例〉

① PRINT LEN ("ABCDEFG")　　「ABCDEFG」という文字列の総文字数「7」を画面に表示しています。

② A＝LEN（"トウキョウ"）

「トウキョウ」という文字列の総文字数「5」を変数Aに代入しています。

③ N$＝"チョコレート"＋CHR$(13)

B＝LEN(N$)

「チョコレート$^{C}{}_{R}$」という文字列の総文字数「7」を変数Bに代入しています。なお，「$^{C}{}_{R}$」は文字コード「13」のコントロールコードを表します。

⑽ INSTR関数

INSTR関数は，文字列の中から特定の文字を探し出し，その文字の含まれる位置を求めます。

(INSTR関数の説明)

〈一般形式〉

INSTR（数式，文字列1，文字列2）

・文字列1の中から文字列2で指定した文字列を探し，見つかれば発見した位置を求めます。見つからなければ結果は「0」になります。

・数式には，文字列1の中の探し始める位置を指定します。

・数式を省略すると，文字列1の先頭の位置から探し始めます。

・文字列2にヌルストリングを指定すると，数式と同じ値を与えます。

〈書き方例〉

① PRINT INSTR(3，"ABCDEFG"，"E")

「ABCDEFG」という文字列の3番目の文字から「E」を探し始め，発見した位置「5」を画面に表示します。

② PRINT INSTR(4，"トウキョウ"，"ト")

「トウキョウ」という文字列の4番目の文字から「ト」を探し始め，発見できなかったので「0」を画面に表示します。

③ M$＝"オオサカ"：X$＝"サカ"

A＝INSTR(M$，X$)

「オオサカ」という文字列の先頭から「サカ」を探し始め，発見した位置「3」を変数Aに代入します。

④ N$＝"フクオカ"：Y$＝" "：J＝2

B＝INSTR(J，N$，Y$)

文字列2にヌルストリングを指定したので，Jの値の「2」を変数Bに代入します。

(11) ASC関数

ASC関数は，CHR$関数と逆の機能を持っていて，文字に付けられているコードを求めます。

(ASC関数の説明)

〈一般形式〉

ASC（文字列）

・文字列の先頭の文字に付けられている文字コードを求めます。

〈書き方例〉

①	PRINT ASC("A")	文字「A」に付けられている文字コード「65」を画面に表示します。
②	X$＝"コウベ" N＝ASC(X$)	「コウベ」の先頭の文字「コ」に付けられている文字コード「186」を変数Nに代入します。
③	Y$＝CHR$(13) M＝ASC(Y$)	「C_R（リターンコード）」に付けられている文字コード「13」を変数Mに代入します。

(12) HEX$関数

HEX$関数は，10進数を16進の文字に変換する関数です。

(HEX$関数の説明)

〈一般形式〉

HEX$（数式）

・数式の値を16進数に変換し，その文字列を求めます。

・数式には，－32768～32767の範囲値を指定することができます。

・求まる文字列は，0～9の数字とA～Fの英文字で表す16進数になります。

〈書き方例〉

①	A$＝HEX$(75)	10進数の「75」を16進数の「4B」に変換して，変数A$に代入します。
②	N＝28460 PRINT HEX$(N)	10進数の「28460」を16進数の「6F2C」に変換して，画面に表示します。

(13) TIME＄関数

TIME$関数は，内蔵時計の持つ時刻を求めたり，内蔵時計に時刻をセットしたりする関数です。

(TIME$関数の説明)

〈一般形式〉

TIME$

・内蔵時計が持っている現在の時刻を求めます。

・TIME$関数は，次の時刻を持っています。

hh：mm：ss

・「hh」は「時」を表し，00～23の値が入っています。

・「mm」は「分」を表し，00～59の値が入っています。

・「ss」は「秒」を表し，00～59の値が入っています。

〈一般形式2〉

TIME$＝文字式

・内蔵時計に文字式の値で時刻をセットします。

・文字式は，次の形式で指定します。

hh：mm：ss

〈書き方例〉

①	PRINT　TIME$	現在の時刻を画面に表示します。
②	T$＝TIME$	現在の時刻を変数T$に代入します。
③	TIME$＝"09：25：40"	内蔵時計に「9時25分40秒」をセットします。
④	INPUT　"ジコク"；A$ TIME$＝A$	キーボードから時刻を「hh：mm：ss」の形式で入力し，その時刻を内蔵時計にセットします。

(14) DATE＄関数

DATE$関数は，内蔵カレンダーが持つ日付を求めたり，内蔵カレンダーに日付をセットしたりする関数です。

(DATE$関数の説明)

〈一般形式1〉

DATE$

・内蔵カレンダーが持っている現在の日付を求めます。

・DATE$関数は，次の形式で日付を持っています。

YY／MM／DD

・「YY」は「年」を表します。

・「MM」は「月」を表し，01～12の値が入っています。

・「DD」は「日」を表し，01～31の値が入っています。

〈一般形式 2〉

DATE$＝文字式

・内蔵カレンダーに文字式の値で日付をセットします。

・文字式は，次の形式で日付をセットします。

YY／MM／DD

・電源を切ったりリセットボタンを押したりすると，年を表す「YY」の値が初期値の「84」にイニシャライズされます。

〈書き方例〉

①	PRINT DATE$	現在の日付を画面に表示します。
②	D$＝DATE$	現在の日付を変数D$に代入します。
③	DATE$＝"84／10／29"	内蔵カレンダーに「84年10月29日」をセットします。
④	INPUT "ヒヅケ"；B$ DATE$＝B$	キーボードから日付を「YY／MM／DD」の形式で入力し，その日付を内蔵カレンダーにセットします。

付.3　キャラクタコード表

このキャラクタコード表は16進数での表現になっていて，「上位４ビット」を先に「下位４ビット」を後に書きます。たとえば，「K」を示すには，「&H4B」と表します。

上位４ビット→　下位４ビット↓

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		D_E		0	@	P		p				ー	タ	ミ		
1	S_H	D_1	!	1	A	Q	a	q			。	ア	チ	ム		円
2	S_X	D_2	"	2	B	R	b	r			「	イ	ツ	メ		年
3	E_X	D_3	#	3	C	S	c	s			」	ウ	テ	モ		月
4	E_T	D_4	$	4	D	T	d	t			、	エ	ト	ヤ		日
5	E_Q	N_K	%	5	E	U	e	u			・	オ	ナ	ユ		時
6	A_K	S_N	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		分
7	B_L	E_B	'	7	G	W	g	w			ァ	キ	ヌ	ラ		秒
8	B_S	C_N	(	8	H	X	h	x			ィ	ク	ネ	リ	♠	
9	H_T	E_M	)	9	I	Y	i	y			ゥ	ケ	ノ	ル	♥	
A	L_F	S_B	*	:	J	Z	j	z			ェ	コ	ハ	レ	♦	
B	H_M	E_C	+	;	K	[	k	{			ォ	サ	ヒ	ロ	♣	
C	C_L	→	,	<	L	¥	l	¦			ャ	シ	フ	ワ	●	
D	C_R	←	—	=	M	]	m	}			ュ	ス	ヘ	ン	○	
E	S_O	↑	.	>	N		n	~			ョ	セ	ホ	゛		
F	S_I	↓	/	?	O	_	o				ッ	ソ	マ	゜		

付.4 漢字コード一覧表

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
記号	212			、	。	，	．	・	：	；	？	！	゛	゜	´	｀	¨
	213	＾	￣	＿	ヽ	ヾ	ゝ	ゞ	〃	仝	々	〆	〇	ー	―	‐	／
	214	＼	～	‖	｜	…	‥	‘	’	“	”	（	）	〔	〕	［	］
	215	｛	｝	〈	〉	《	》	「	」	『	』	【	】	＋	－	±	×
	216	÷	＝	≠	＜	＞	≦	≧	∞	∴	♂	♀	°	′	″	℃	￥
	217	＄	¢	£	％	＃	＆	＊	＠	§	☆	★	○	●	◎	◇	
	222		◆	□	■	△	▲	▽	▼	※	〒	→	←	↑	↓	〓	
英・数字	233	0	1	2	3	4	5	6	7	8	9						
	234		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	235	P	Q	R	S	T	U	V	W	X	Y	Z					
	236		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	237	p	q	r	s	t	u	v	w	x	y	z					
ひらがな	242		ぁ	あ	ぃ	い	ぅ	う	ぇ	え	ぉ	お	か	が	き	ぎ	く
	243	ぐ	け	げ	こ	ご	さ	ざ	し	じ	す	ず	せ	ぜ	そ	ぞ	た
	244	だ	ち	ぢ	っ	つ	づ	て	で	と	ど	な	に	ぬ	ね	の	は
	245	ば	ぱ	ひ	び	ぴ	ふ	ぶ	ぷ	へ	べ	ぺ	ほ	ぼ	ぽ	ま	み
	246	む	め	も	ゃ	や	ゅ	ゆ	ょ	よ	ら	り	る	れ	ろ	ゎ	わ
	247	ゐ	ゑ	を	ん												
カタカナ	252		ァ	ア	ィ	イ	ゥ	ウ	ェ	エ	ォ	オ	カ	ガ	キ	ギ	ク
	253	グ	ケ	ゲ	コ	ゴ	サ	ザ	シ	ジ	ス	ズ	セ	ゼ	ソ	ゾ	タ
	254	ダ	チ	ヂ	ッ	ツ	ヅ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ
	255	バ	パ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ベ	ペ	ホ	ボ	ポ	マ	ミ
	256	ム	メ	モ	ャ	ヤ	ュ	ユ	ョ	ヨ	ラ	リ	ル	レ	ロ	ヮ	ワ
	257	ヰ	ヱ	ヲ	ン	ヴ	ヵ	ヶ									
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
半角文字	002		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
	003	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
	004	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	005	P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	^	—
	006		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	007	p	q	r	s	t	u	v	w	x	y	z	{	¦	}	~	
	008		。	「	」	、	・	を	ぁ	ぃ	ぅ	ぇ	ぉ	ゃ	ゅ	ょ	っ
	009	ー	あ	い	う	え	お	か	き	く	け	こ	さ	し	す	せ	そ
	00A		。	「	」		・	ヲ	ァ	ィ	ゥ	ェ	ォ	ャ	ュ	ョ	ッ
	00B	ー	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
	00C	タ	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
	00D	ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	゛	゜
	00E	た	ち	つ	て	と	な	に	ぬ	ね	の	は	ひ	ふ	へ	ほ	ま
	00F	み	む	め	も	や	ゆ	よ	ら	り	る	れ	ろ	わ	ん	゛	゜
1/4角文字	010		SH	SX	EX	ET	EQ	AK	BL	BS	HT	LF	HM	CL	CR	SO	SI
	011	DE	D1	D2	D3	D4	NK	SN	EB	CN	EM	SB	EC	→	←	↑	↓
	012		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
	013	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
	014	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	015	P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	∧	—
	016		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	017	p	q	r	s	t	u	v	w	x	y	z	{	¦	}	~	
	018	▁	▂	▃	▄	▅	▆	▇	█	▏	▎	▍	▌	▋	▊	▉	┼
	019	┴	┬	┤	├	▔	─	│	▕	┌	┐	└	┘	╭	╮	╰	╯
	01A		。	「	」	、	・	ヲ	ァ	ィ	ゥ	ェ	ォ	ャ	ュ	ョ	ッ
	01B	ー	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
	01C	タ	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
	01D	ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	゛	゜
	01E	═	╞	╪	╡	◢	◣	◥	◤	♠	♥	♦	♣	●	○	╱	╲
	01F	×	円	年	月	日	時	分	秒								
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ギリシア文字	262		Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο
	263	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω							
	264		α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
	265	π	ρ	σ	τ	υ	φ	χ	ψ	ω							
ロシア文字	272		А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н
	273	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
	274	Ю	Я														
	275		а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н
	276	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э
	277	ю	я														
ア	302		亜	唖	娃	阿	哀	愛	挨	姶	逢	葵	茜	穐	悪	握	渥
	303	旭	葦	芦	鯵	梓	圧	斡	扱	宛	姐	虻	飴	絢	綾	鮎	或
	304	粟	袷	安	庵	按	暗	案	闇	鞍	杏						
イ	304											以	伊	位	依	偉	囲
	305	夷	委	威	尉	惟	意	慰	易	椅	為	畏	異	移	維	緯	胃
	306	萎	衣	謂	違	遺	医	井	亥	域	育	郁	磯	一	壱	溢	逸
	307	稲	茨	芋	鰯	允	印	咽	員	因	姻	引	飲	淫	胤	蔭	
	312		院	陰	隠	韻	吋										
ウ	312							右	宇	烏	羽	迂	雨	卯	鵜	窺	丑
	313	碓	臼	渦	嘘	唄	欝	蔚	鰻	姥	厩	浦	瓜	閏	噂	云	運
	314	雲															
エ	314		荏	餌	叡	営	嬰	影	映	曳	栄	永	泳	洩	瑛	盈	穎
	315	頴	英	衛	詠	鋭	液	疫	益	駅	悦	謁	越	閲	榎	厭	円
	316	園	堰	奄	宴	延	怨	掩	援	沿	演	炎	焔	煙	燕	猿	縁
	317	艶	苑	薗	遠	鉛	鴛	塩									
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
オ	317								於	汚	甥	凹	央	奥	往	応	
	322		押	旺	横	欧	殴	王	翁	襖	鴬	鴎	黄	岡	沖	荻	億
	323	屋	憶	臆	桶	牡	乙	俺	卸	恩	温	穏	音				
カ	323													下	化	仮	何
	324	伽	価	佳	加	可	嘉	夏	嫁	家	寡	科	暇	果	架	歌	河
	325	火	珂	禍	禾	稼	箇	花	苛	茄	荷	華	菓	蝦	課	嘩	貨
	326	迦	過	霞	蚊	俄	峨	我	牙	画	臥	芽	蛾	賀	雅	餓	駕
	327	介	会	解	回	塊	壊	廻	快	怪	悔	恢	懐	戒	拐	改	
	332		魁	晦	械	海	灰	界	皆	絵	芥	蟹	開	階	貝	凱	劾
	333	外	咳	害	崖	慨	概	涯	碍	蓋	街	該	鎧	骸	浬	馨	蛙
	334	垣	柿	蛎	鈎	劃	嚇	各	廓	拡	撹	格	核	殻	獲	確	穫
	335	覚	角	赫	較	郭	閣	隔	革	学	岳	楽	額	顎	掛	笠	樫
	336	橿	梶	鰍	潟	割	喝	恰	括	活	渇	滑	葛	褐	轄	且	鰹
	337	叶	椛	樺	鞄	株	兜	竃	蒲	釜	鎌	噛	鴨	栢	茅	萱	
	342		粥	刈	苅	瓦	乾	侃	冠	寒	刊	勘	勧	巻	喚	堪	姦
	343	完	官	寛	干	幹	患	感	慣	憾	換	敢	柑	桓	棺	款	歓
	344	汗	漢	澗	潅	環	甘	監	看	竿	管	簡	緩	缶	翰	肝	艦
	345	莞	観	諌	貫	還	鑑	間	閑	関	陥	韓	館	舘	丸	含	岸
	346	巌	玩	癌	眼	岩	翫	贋	雁	頑	顔	願					
キ	346												企	伎	危	喜	器
	347	基	奇	嬉	寄	岐	希	幾	忌	揮	机	旗	既	期	棋	棄	
	352		機	帰	毅	気	汽	畿	祈	季	稀	紀	徽	規	記	貴	起
	353	軌	輝	飢	騎	鬼	亀	偽	儀	妓	宜	戯	技	擬	欺	犠	疑
	354	祇	義	蟻	誼	議	掬	菊	鞠	吉	吃	喫	桔	橘	詰	砧	杵
	355	黍	却	客	脚	虐	逆	丘	久	仇	休	及	吸	宮	弓	急	救
	356	朽	求	汲	泣	灸	球	究	窮	笈	級	糾	給	旧	牛	去	居
	357	巨	拒	拠	挙	渠	虚	許	距	鋸	漁	禦	魚	亨	享	京	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	362		供	侠	僑	兇	競	共	凶	協	匡	卿	叫	喬	境	峡	強
	363	彊	怯	恐	恭	挟	教	橋	況	狂	狭	矯	胸	脅	興	蕎	郷
	364	鏡	響	饗	驚	仰	凝	尭	暁	業	局	曲	極	玉	桐	粁	僅
	365	勤	均	巾	錦	斤	欣	欽	琴	禁	禽	筋	緊	芹	菌	衿	襟
	366	謹	近	金	吟	銀											
ク	366						九	倶	句	区	狗	玖	矩	苦	躯	駆	駈
	367	駒	具	愚	虞	喰	空	偶	寓	遇	隅	串	櫛	釧	屑	屈	
	372		掘	窟	沓	靴	轡	窪	熊	隈	粂	栗	繰	桑	鍬	勲	君
	373	薫	訓	群	軍	郡											
ケ	373						卦	袈	祁	係	傾	刑	兄	啓	圭	珪	型
	374	契	形	径	恵	慶	慧	憩	掲	携	敬	景	桂	渓	畦	稽	系
	375	経	継	繋	罫	茎	荊	蛍	計	詣	警	軽	頚	鶏	芸	迎	鯨
	376	劇	戟	撃	激	隙	桁	傑	欠	決	潔	穴	結	血	訣	月	件
	377	倹	倦	健	兼	券	剣	喧	圏	堅	嫌	建	憲	懸	拳	捲	
	382		検	権	牽	犬	献	研	硯	絹	県	肩	見	謙	賢	軒	遣
	383	鍵	険	顕	験	鹸	元	原	厳	幻	弦	減	源	玄	現	絃	舷
	384	言	諺	限													
コ	384				乎	個	古	呼	固	姑	孤	己	庫	弧	戸	故	枯
	385	湖	狐	糊	袴	股	胡	菰	虎	誇	跨	鈷	雇	顧	鼓	五	互
	386	伍	午	呉	吾	娯	後	御	悟	梧	檎	瑚	碁	語	誤	護	醐
	387	乞	鯉	交	佼	侯	候	倖	光	公	功	効	勾	厚	口	向	
	392		后	喉	坑	垢	好	孔	孝	宏	工	巧	巷	幸	広	庚	康
	393	弘	恒	慌	抗	拘	控	攻	昂	晃	更	杭	校	梗	構	江	洪
	394	浩	港	溝	甲	皇	硬	稿	糠	紅	紘	絞	綱	耕	考	肯	肱
	395	腔	膏	航	荒	行	衡	講	貢	購	郊	酵	鉱	礦	鋼	閤	降
	396	項	香	高	鴻	剛	劫	号	合	壕	拷	濠	豪	轟	麹	克	刻
	397	告	国	穀	酷	鵠	黒	獄	漉	腰	甑	忽	惚	骨	狛	込	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	3A2		此	頃	今	困	坤	墾	婚	恨	懇	昏	昆	根	梱	混	痕
	3A3	紺	艮	魂													
サ	3A3				些	佐	叉	唆	嵯	左	差	査	沙	瑳	砂	詐	鎖
	3A4	裟	坐	座	挫	債	催	再	最	哉	塞	妻	宰	彩	才	採	栽
	3A5	歳	済	災	采	犀	砕	砦	祭	斎	細	菜	裁	載	際	剤	在
	3A6	材	罪	財	冴	坂	阪	堺	榊	肴	咲	崎	埼	碕	鷺	作	削
	3A7	咋	搾	昨	朔	柵	窄	策	索	錯	桜	鮭	笹	匙	冊	刷	
	3B2		察	拶	撮	擦	札	殺	薩	雑	皐	鯖	捌	錆	鮫	皿	晒
	3B3	三	傘	参	山	惨	撒	散	桟	燦	珊	産	算	纂	蚕	讃	賛
	3B4	酸	餐	斬	暫	残											
シ	3B4						仕	仔	伺	使	刺	司	史	嗣	四	士	始
	3B5	姉	姿	子	屍	市	師	志	思	指	支	孜	斯	施	旨	枝	止
	3B6	死	氏	獅	祉	私	糸	紙	紫	肢	脂	至	視	詞	詩	試	誌
	3B7	諮	資	賜	雌	飼	歯	事	似	侍	児	字	寺	慈	持	時	
	3C2		次	滋	治	爾	璽	痔	磁	示	而	耳	自	蒔	辞	汐	鹿
	3C3	式	識	鴫	竺	軸	宍	雫	七	叱	執	失	嫉	室	悉	湿	漆
	3C4	疾	質	実	蔀	篠	偲	柴	芝	屢	蘂	縞	舎	写	射	捨	赦
	3C5	斜	煮	社	紗	者	謝	車	遮	蛇	邪	借	勺	尺	杓	灼	爵
	3C6	酌	釈	錫	若	寂	弱	惹	主	取	守	手	朱	殊	狩	珠	種
	3C7	腫	趣	酒	首	儒	受	呪	寿	授	樹	綬	需	囚	収	周	
	3D2		宗	就	州	修	愁	拾	洲	秀	秋	終	繍	習	臭	舟	蒐
	3D3	衆	襲	讐	蹴	輯	週	酋	酬	集	醜	什	住	充	十	従	戎
	3D4	柔	汁	渋	獣	縦	重	銃	叔	夙	宿	淑	祝	縮	粛	塾	熟
	3D5	出	術	述	俊	峻	春	瞬	竣	舜	駿	准	循	旬	楯	殉	淳
	3D6	準	潤	盾	純	巡	遵	醇	順	処	初	所	暑	曙	渚	庶	緒
	3D7	署	書	薯	藷	諸	助	叙	女	序	徐	恕	鋤	除	傷	償	
	3E2		勝	匠	升	召	哨	商	唱	嘗	奨	妾	娼	宵	将	小	少
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
シ	3E3	尚	庄	床	廠	彰	承	抄	招	掌	捷	昇	昌	昭	晶	松	梢
	3E4	樟	樵	沼	消	渉	湘	焼	焦	照	症	省	硝	礁	祥	称	章
	3E5	笑	粧	紹	肖	菖	蒋	蕉	衝	裳	訟	証	詔	詳	象	賞	醤
	3E6	鉦	鍾	鐘	障	鞘	上	丈	丞	乗	冗	剰	城	場	壌	嬢	常
	3E7	情	擾	条	杖	浄	状	畳	穣	蒸	譲	醸	錠	嘱	埴	飾	
	3F2		拭	植	殖	燭	織	職	色	触	食	蝕	辱	尻	伸	信	侵
	3F3	唇	娠	寝	審	心	慎	振	新	晋	森	榛	浸	深	申	疹	真
	3F4	神	秦	紳	臣	芯	薪	親	診	身	辛	進	針	震	人	仁	刃
	3F5	塵	壬	尋	甚	尽	腎	訊	迅	陣	靭						
ス	3F5											笥	諏	須	酢	図	厨
	3F6	逗	吹	垂	帥	推	水	炊	睡	粋	翠	衰	遂	酔	錐	錘	随
	3F7	瑞	髄	崇	嵩	数	枢	趨	雛	据	杉	椙	菅	頗	雀	裾	
	402		澄	摺	寸												
セ	402					世	瀬	畝	是	凄	制	勢	姓	征	性	成	政
	403	整	星	晴	棲	栖	正	清	牲	生	盛	精	聖	声	製	西	誠
	404	誓	請	逝	醒	青	静	斉	税	脆	隻	席	惜	戚	斥	昔	析
	405	石	積	籍	績	脊	責	赤	跡	蹟	碩	切	拙	接	摂	折	設
	406	窃	節	説	雪	絶	舌	蝉	仙	先	千	占	宣	専	尖	川	戦
	407	扇	撰	栓	栴	泉	浅	洗	染	潜	煎	煽	旋	穿	箭	線	
	412		繊	羨	腺	舛	船	薦	詮	賎	践	選	遷	銭	銑	閃	鮮
	413	前	善	漸	然	全	禅	繕	膳	糎							
ソ	413										噌	塑	岨	措	曾	曽	楚
	414	狙	疏	疎	礎	祖	租	粗	素	組	蘇	訴	阻	遡	鼠	僧	創
	415	双	叢	倉	喪	壮	奏	爽	宋	層	匝	惣	想	捜	掃	挿	掻
	416	操	早	曹	巣	槍	槽	漕	燥	争	痩	相	窓	糟	総	綜	聡
	417	草	荘	葬	蒼	藻	装	走	送	遭	鎗	霜	騒	像	増	憎	

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	422		臓	蔵	贈	造	促	側	則	即	息	捉	束	測	足	速	俗
	423	属	賊	族	続	卒	袖	其	揃	存	孫	尊	損	村	遜		
タ	423															他	多
	424	太	汰	詑	唾	堕	妥	惰	打	柁	舵	楕	陀	駄	騨	体	堆
	425	対	耐	岱	帯	待	怠	態	戴	替	泰	滞	胎	腿	苔	袋	貸
	426	退	逮	隊	黛	鯛	代	台	大	第	醍	題	鷹	滝	瀧	卓	啄
	427	宅	托	択	拓	沢	濯	琢	託	鐸	濁	諾	茸	凧	蛸	只	
	432		叩	但	達	辰	奪	脱	巽	竪	辿	棚	谷	狸	鱈	樽	誰
	433	丹	単	嘆	坦	担	探	旦	歎	淡	湛	炭	短	端	箪	綻	耽
	434	胆	蛋	誕	鍛	団	壇	弾	断	暖	檀	段	男	談			
チ	434														値	知	地
	435	弛	恥	智	池	痴	稚	置	致	蜘	遅	馳	築	畜	竹	筑	蓄
	436	逐	秩	窒	茶	嫡	着	中	仲	宙	忠	抽	昼	柱	注	虫	衷
	437	註	酎	鋳	駐	樗	瀦	猪	苧	著	貯	丁	兆	凋	喋	寵	
	442		帖	帳	庁	弔	張	彫	徴	懲	挑	暢	朝	潮	牒	町	眺
	443	聴	脹	腸	蝶	調	諜	超	跳	銚	長	頂	鳥	勅	捗	直	朕
	444	沈	珍	賃	鎮	陳											
ツ	444						津	墜	椎	槌	追	鎚	痛	通	塚	栂	掴
	445	槻	佃	漬	柘	辻	蔦	綴	鍔	椿	潰	坪	壷	嬬	紬	爪	吊
	446	釣	鶴														
テ	446			亭	低	停	偵	剃	貞	呈	堤	定	帝	底	庭	廷	弟
	447	悌	抵	挺	提	梯	汀	碇	禎	程	締	艇	訂	諦	蹄	逓	
	452		邸	鄭	釘	鼎	泥	摘	擢	敵	滴	的	笛	適	鏑	溺	哲
	453	徹	撤	轍	迭	鉄	典	填	天	展	店	添	纏	甜	貼	転	顛
	454	点	伝	殿	澱	田	電										

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ト	454							兎	吐	堵	塗	妬	屠	徒	斗	杜	渡
	455	登	菟	賭	途	都	鍍	砥	礪	努	度	土	奴	怒	倒	党	冬
	456	凍	刀	唐	塔	塘	套	宕	島	嶋	悼	投	搭	東	桃	檮	棟
	457	盗	淘	湯	濤	灯	燈	当	痘	禱	等	答	筒	糖	統	到	
	462		董	蕩	藤	討	謄	豆	踏	逃	透	鐙	陶	頭	騰	闘	働
	463	動	同	堂	導	憧	撞	洞	瞳	童	胴	萄	道	銅	峠	鴇	匿
	464	得	徳	瀆	特	督	禿	篤	毒	独	読	栃	橡	凸	突	椴	届
	465	鳶	苫	寅	酉	瀞	噸	屯	惇	敦	沌	豚	遁	頓	呑	曇	鈍
ナ	466	奈	那	内	乍	凪	薙	謎	灘	捺	鍋	楢	馴	縄	畷	南	楠
	467	軟	難	汝													
ニ	467				二	尼	弐	迩	匂	賑	肉	虹	廿	日	乳	入	
	472		如	尿	韮	任	妊	忍	認								
ヌ	472									濡							
ネ	472										禰	祢	寧	葱	猫	熱	年
	473	念	捻	撚	燃	粘											
ノ	473						乃	廼	之	埜	嚢	悩	濃	納	能	脳	膿
	474	農	覗	蚤													
ハ	474				巴	把	播	覇	杷	波	派	琶	破	婆	罵	芭	馬
	475	俳	廃	拝	排	敗	杯	盃	牌	背	肺	輩	配	倍	培	媒	梅
	476	楳	煤	狽	買	売	賠	陪	這	蝿	秤	矧	萩	伯	剥	博	拍
	477	柏	泊	白	箔	粕	舶	薄	迫	曝	漠	爆	縛	莫	駁	麦	
	482		函	箱	硲	箸	肇	筈	櫨	幡	肌	畑	畠	八	鉢	溌	発
	483	醗	髪	伐	罰	抜	筏	閥	鳩	噺	塙	蛤	隼	伴	判	半	反
	484	叛	帆	搬	斑	板	氾	汎	版	犯	班	畔	繁	般	藩	販	範
	485	釆	煩	頒	飯	挽	晩	番	盤	磐	蕃	蛮					
ヒ	485												匪	卑	否	妃	庇
	486	彼	悲	扉	批	披	斐	比	泌	疲	皮	碑	秘	緋	罷	肥	被
	487	誹	費	避	非	飛	樋	簸	備	尾	微	枇	毘	琵	眉	美	
	492		鼻	柊	稗	匹	疋	髭	彦	膝	菱	肘	弼	必	畢	筆	逼
	493	桧	姫	媛	紐	百	謬	俵	彪	標	氷	漂	瓢	票	表	評	豹
	494	廟	描	病	秒	苗	錨	鋲	蒜	蛭	鰭	品	彬	斌	浜	瀕	貧
	495	賓	頻	敏	瓶												
フ	495					不	付	埠	夫	婦	富	冨	布	府	怖	扶	敷
	496	斧	普	浮	父	符	腐	膚	芙	譜	負	賦	赴	阜	附	侮	撫
	497	武	舞	葡	蕪	部	封	楓	風	葺	蕗	伏	副	復	幅	服	
	4A2		福	腹	複	覆	淵	弗	払	沸	仏	物	鮒	分	吻	噴	墳
	4A3	憤	扮	焚	奮	粉	糞	紛	雰	文	聞						
ヘ	4A3											丙	併	兵	塀	幣	平
	4A4	弊	柄	並	蔽	閉	陛	米	頁	僻	壁	癖	碧	別	瞥	蔑	箆
	4A5	偏	変	片	篇	編	辺	返	遍	便	勉	娩	弁	鞭			
ホ	4A5														保	舗	鋪
	4A6	圃	捕	歩	甫	補	輔	穂	募	墓	慕	戊	暮	母	簿	菩	倣
	4A7	俸	包	呆	報	奉	宝	峰	峯	崩	庖	抱	捧	放	方	朋	
	4B2		法	泡	烹	砲	縫	胞	芳	萌	蓬	蜂	褒	訪	豊	邦	鋒
	4B3	飽	鳳	鵬	乏	亡	傍	剖	坊	妨	帽	忘	忙	房	暴	望	某
	4B4	棒	冒	紡	肪	膨	謀	貌	貿	鉾	防	吠	頬	北	僕	卜	墨
	4B5	撲	朴	牧	睦	穆	釦	勃	没	殆	堀	幌	奔	本	翻	凡	盆

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
マ	4B6	摩	磨	魔	麻	埋	妹	昧	枚	毎	哩	槙	幕	膜	枕	鮪	柾
	4B7	鱒	桝	亦	俣	又	抹	末	沫	迄	儘	繭	麿	万	慢	満	
	4C2		漫	蔓													
ミ	4C2				味	未	魅	巳	箕	岬	密	蜜	湊	蓑	稔	脈	妙
	4C3	粍	民	眠													
ム	4C3				務	夢	無	牟	矛	霧	鵡	椋	婿	娘			
メ	4C3														冥	名	命
	4C4	明	盟	迷	銘	鳴	姪	牝	滅	免	棉	綿	緬	面	麺		
モ	4C4															摸	模
	4C5	茂	妄	孟	毛	猛	盲	網	耗	蒙	儲	木	黙	目	杢	勿	餅
	4C6	尤	戻	籾	貰	問	悶	紋	門	匁							
ヤ	4C6										也	冶	夜	爺	耶	野	弥
	4C7	矢	厄	役	約	薬	訳	躍	靖	柳	薮	鑓					
ユ	4C7												愉	愈	油	癒	
	4D2		諭	輸	唯	佑	優	勇	友	宥	幽	悠	憂	揖	有	柚	湧
	4D3	涌	猶	猷	由	祐	裕	誘	遊	邑	郵	雄	融	夕			
ヨ	4D3														予	余	与
	4D4	誉	輿	預	傭	幼	妖	容	庸	揚	揺	擁	曜	楊	様	洋	溶
	4D5	熔	用	窯	羊	耀	葉	蓉	要	謡	踊	遙	陽	養	慾	抑	欲
	4D6	沃	浴	翌	翼	淀											
ラ	4D6						羅	螺	裸	来	莱	頼	雷	洛	絡	落	酪
	4D7	乱	卵	嵐	欄	濫	藍	蘭	覧								
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
リ	4D7									利	吏	履	李	梨	理	璃	
	4E2		痢	裏	裡	里	離	陸	律	率	立	葎	掠	略	劉	流	溜
	4E3	琉	留	硫	粒	隆	竜	龍	侶	慮	旅	虜	了	亮	僚	両	凌
	4E4	寮	料	梁	涼	猟	療	瞭	稜	糧	良	諒	遼	量	陵	領	力
	4E5	緑	倫	厘	林	淋	燐	琳	臨	輪	隣	鱗	麟				
ル	4E5													瑠	塁	涙	累
	4E6	類															
レ	4E6		令	伶	例	冷	励	嶺	怜	玲	礼	苓	鈴	隷	零	霊	麗
	4E7	齢	暦	歴	列	劣	烈	裂	廉	恋	憐	漣	煉	簾	練	聯	
	4F2		蓮	連	錬												
ロ	4F2					呂	魯	櫓	炉	賂	路	露	労	婁	廊	弄	朗
	4F3	楼	榔	浪	漏	牢	狼	篭	老	聾	蝋	郎	六	麓	禄	肋	録
	4F4	論															
ワ	4F4		倭	和	話	歪	賄	脇	惑	枠	鷲	亙	亘	鰐	詫	藁	蕨
	4F5	椀	湾	碗	腕												
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

付.5　エラーコード一覧表

注意：エラーメッセージの欄で，フルセンテンスで示されているものは，N_{88} Disk-BASIC または N - BASIC のメッセージであり，カッコ内に示されている省略形のメッセージは N_{88} - BASIC のものです．エラーコードの欄は上段が N - BASIC，下段が N_{88} - BASIC のコードを表わしています．“—”で示されているのは，存在しないエラーです．

エラーメッセージ	コード	意　　味
Bad allocation table	N 64 N88 69	ディスケット内部のFATが壊れている．
Bad drive number	N 65 N88 70	ドライブ指定が誤っている．システムにつながっていないドライブを指定した．
Bad File Data	N 25 —	ファイル上にあるデータの形式がまちがっている．
Bad file name (? NM Error)	N 62 N88 56	ファイル名の指定がまちがっている．
Bad file number (? BN Error)	N 52 N88 52	オープンしていないファイルや起動時に指定していないファイルを参照した．
Bad file mode	N 54 —	シーケンシャルでオープンしたファイルに対してランダムアクセスをしようとした．またはその逆．
Bad record number	N 63 —	ランダムアクセスファイルで許容されるレコードナンバーから外れている．
Bad track / sector	N 66 N88 71	トラック，セクタ番号の指定が誤っている．（DSKO$，DSKI$）
Can't continue (? CN Error)	N 17 N88 17	CONT命令によって実行を再開することができない（ポインタの値が壊されている）．
Communications Buffer Overflow	N 27 —	周辺機器との入出力のためのバッファがオーバーフローした．（N88 -BASIC では Line buffer overflow）
Deleted record	N 67 N88 72	消去済みのレコードをアクセスしようとした．
Direct Statment in file (? DS Error)	N 63 N88 57	アスキー形式のファイルを読み込む際にダイレクトステートメントが存在した．
Disk already mounted	N 59 N88 67	mount されているディスクに対して mount を行った．
Disk BASIC Feature	N 26 —	ディスクが接続されていないとき，ディスクBASICの命令を実行した．

エラーメッセージ	コード	意　　味
Disk full	N 60 N88 68	ディスク上に書き込むスペースがないのに書き込もうとした.
Disk not mounted	N 56 N88 63	mountされていないディスクに対してアクセスした.
Disk I/O Error	N 57 N88 64	ディスクとの入出力中にエラーが発生した. 致命的なエラーであり回復させることはできない.
Disk offline	N 73 N88 62	入出力の可能な状態でないディスクをアクセスしようとした.
Division by Zero (? /0 Error)	N 11 N88 11	0による割り算が実行されて，その結果の値がオーバーフローした.
Duplicate Definition (? DD Error)	— N88 10	配列またはユーザー関数を二重定義しようとした. (N-BASICでは，Redimensioned array)
duplicate label (? DU Error)	— N88 31	同じラベル名が二つ以上存在している.
Feature not available (? NA Error)	— N88 33	利用不可能な機能を指定した.
FIELD overflow (? FO Error)	N 50 N88 50	FIELD文において256バイト以上の大きさの領域を指定した.
File already exists	N 58 N88 65	NAME文によって変更しようとしたファイル名が既に存在している。
File already open (? AO Error)	N 55 N88 54	既にオープンされているファイルに対してOPEN，KILLなどを実行した.
File not found (?FF Error)	N 53 N88 53	LOAD，SAVE，KILLなどで現在のディスクに存在しないファイルを指定した.
File not Open (? CF Error)	N 71 N88 60	オープンされていないファイルを参照しようとした.
File write protected	N 72 N88 61	書き込み禁止属性が付けられているファイルに書き込もうとした.
For without NEXT (? FN Error)	— N88 26	FOR～NEXTが正しく対応していない（FORが多い）.
Illegal direct (? ID Error)	N 12 N88 12	ダイレクトモードで使用できない文を実行しようとした.

エラーメッセージ	コード	意　味
Illegal function call (? FC Error)	N 5 N88 5	ステートメントおよび関数において，機能の呼び方が誤っている．
Input past end (? EF Error)	N 61 N88 55	ファイル中のすべてのデータを読み尽くした後に，さらに入力文が実行された．
Internal error (?IE Error)	N 51 N88 51	BASIC 内部にエラーが生じた．
Line buffer overflow (?BO Error)	N 23 N88 23	1行で入力できる文字の範囲を越えて入力が行われた．
Missing operand (? MO Error)	N 22 N88 22	ステートメント中必要なパラメータが指定されていない．
NEXT without FOR (? NF Error)	N 1 N88 1	FOR～NEXT が正しく対応していない，(NEXT が多い)
No RESUME (?NR Error)	N 19 N88 19	エラー処理ルーチンの中に RESUME 文がなく，プログラムの実行が継続できない．
Out of data (? OD Error)	N 4 N88 4	読むべきデータがないのに READ 文が実行された．
Out of memory (? OM Error)	N 7 N88 7	メモリ容量が足りなくなった（プログラムが大きすぎる，またはスタックを消費し尽した）．
Out of string space (? OS Error)	N 14 N88 14	文字列を格納するメモリ領域がなくなった．
Overflow (? OV Error)	N 6 N88 6	演算結果や入力された数値が，許される範囲を超えた．
Port not initialized	N 28 —	インターフェイス用の LSI の機能設定がなされていない．
Position not on Screen	N 24 —	指定したカーソル位置などが，画面の範囲外になっている．
Redimensioned array	N 10 —	配列またはユーザー関数を二重定義しようとした（N88 -BASIC では Dupuplicate Definition）．
Rename across disks	N 68 N88 73	NAME 文において，異なるドライブ間でリネームしようとした．
RESUME without error (? RW Error)	N 20 N88 20	エラー処理ルーチンに制御を移さなかったのに RESUME 文がある．

エラーメッセージ	コード	意　　味
RETURN without GOSUB (? RG Error)	N 3 N88 3	GOSUB～RETURNが正しく対応していない(RETURNが多い).
Sequential after PUT (? SP Error)	N 69 N88 58	PUT文実行後，シーケンシャルファイルをアクセスしようとした.
Sequential I/O Only (?SQ Error)	N 70 N88 59	シーケンシャル入出力以外は行ってはならない.
String formula too complex (? ST Error)	N 16 N88 16	文字式が複雑すぎる. （カッコのネスティングが深すぎるなど）
String too long (? LS Error)	N 15 N88 15	一つの文字変数内の文字数が255文字を超えている.
Subscript out of range (? BS Error) (Bad Subscript)	N 9 N88 9	配列変数の添字の値が規定の範囲から外れている.
Syntax error (? SN Error)	N 2 N88 2	文の記述が文法どおりになっていない.
Tape read error (? TP Error)	N 29 N88 27	テープからの読み込み時にエラーが発生した.
Too many files	N 67 —	ファイルの数が許容される範囲(255個)を超えた.
Type mismatch (? TM Error)	N 13 N88 13	式の左辺・右辺，関数の引数などにおいて変数の型が一致していない.
Undefined label (? UN Error)	— N88 32	定義されていないラベル名で参照した.
undefined line number (? UL Error)	N 8 N88 8	飛び先として必要とされる行番号（GOTO, GOSUBなどの飛び先）が存在しない.
Undefined user function (? UF Error)	N 18 N88 18	DEFFN文によって定義されていないユーザー関数を引用しようとした.
Unprintable error (? UE Error)	N 21 N88 21	メッセージの定義されていないエラー.
WEND without WHILE (? WE Error)	— N88 30	WHILE～WENDが正しく対応していない. （WENDが多い).
WHILE without WEND (? WH Error)	— N88 29	WHILE～WENDが正しく対応していない. （WHILEが多い).

〔索　　引〕

タ 行

ナ 行

ハ 行

マ 行

ヤ 行

ラ 行

〈コマンド・命令・関数〉

〈参考にした図書〉

● PC-8801mkII　BASICリファレンスマニュアル（NEC）

● PC-8801mkII　ユーザーズマニュアル（NEC）

● PC-8800　BASIC入門編（電子開発学園）

● PC-8800　BASIC初級編（電子開発学園）

● PC-8800　BASIC中級編（電子開発学園）

● PC-8001mkII　BASIC　上巻（電子開発学園）

● PC-8001mkII　BASIC　下巻（電子開発学園）

監 修 者 略 歴

工学博士 松 尾 三 郎（まつお さぶろう）

〔略 歴〕

昭和13年３月	京都帝国大学工学部電気工学科卒業
昭和20年９月	逓信省電波局勤務，電波課調査係長
昭和22年３月	京都帝国大学工学部講師（～30年３月）
昭和24年11月	電気通信省電気通信研究所方式実用化部電波課長
昭和28年３月	日本電信電話公社電気通信研究所方式部無線課長（～29年４月）
昭和29年４月	ニッポン放送技術局次長（～34年３月）
昭和32年４月	武蔵工業大学教授（～39年３月）
昭和32年５月	日本電波塔株式会社取締役技術部長（～35年４月）
昭和34年４月	日本技術開発株式会社専務取締役（～39年９月）
昭和37年８月	日本ビジネスオートメーション株式会社取締役副社長（～39年９月）
昭和39年４月	北海道ビジネスオートメーション株式会社取締役社長（～44年６月）
昭和40年５月	日本電子開発株式会社取締役社長
昭和45年４月	電子開発学園学園長
昭和51年４月	郵政省電波技術審議会委員
現 在	株式会社イーディシー取締役社長
	電子開発学園学園長
	日本電子開発株式会社取締役社長
	ソフトウェアコンサルタント株式会社取締役社長
	郵政省電波技術審議会委員

PC-8801mkIIの使い方　上巻　　定価2,300円

昭和59年12月１日　　初版第１刷発行

監　修／松　尾　三　郎

編　集／電子開発学園 出版局

発行者／松　尾　三　郎

発行所／㈱イーディシー

電子開発学園 出版局

〒164 東京都中野区中野5丁目62番1号（ＥＤＣビル）

電話 (03) 319－7101

振替　東京２－54892

印刷・製本所／(株)カントー

電話(03)238-6011

ISBN4-88647-023-8

PC-8801mkⅡの使い方 上巻

ISBN4-88647-023-8 C3055 ¥2300E

定価 2,300円